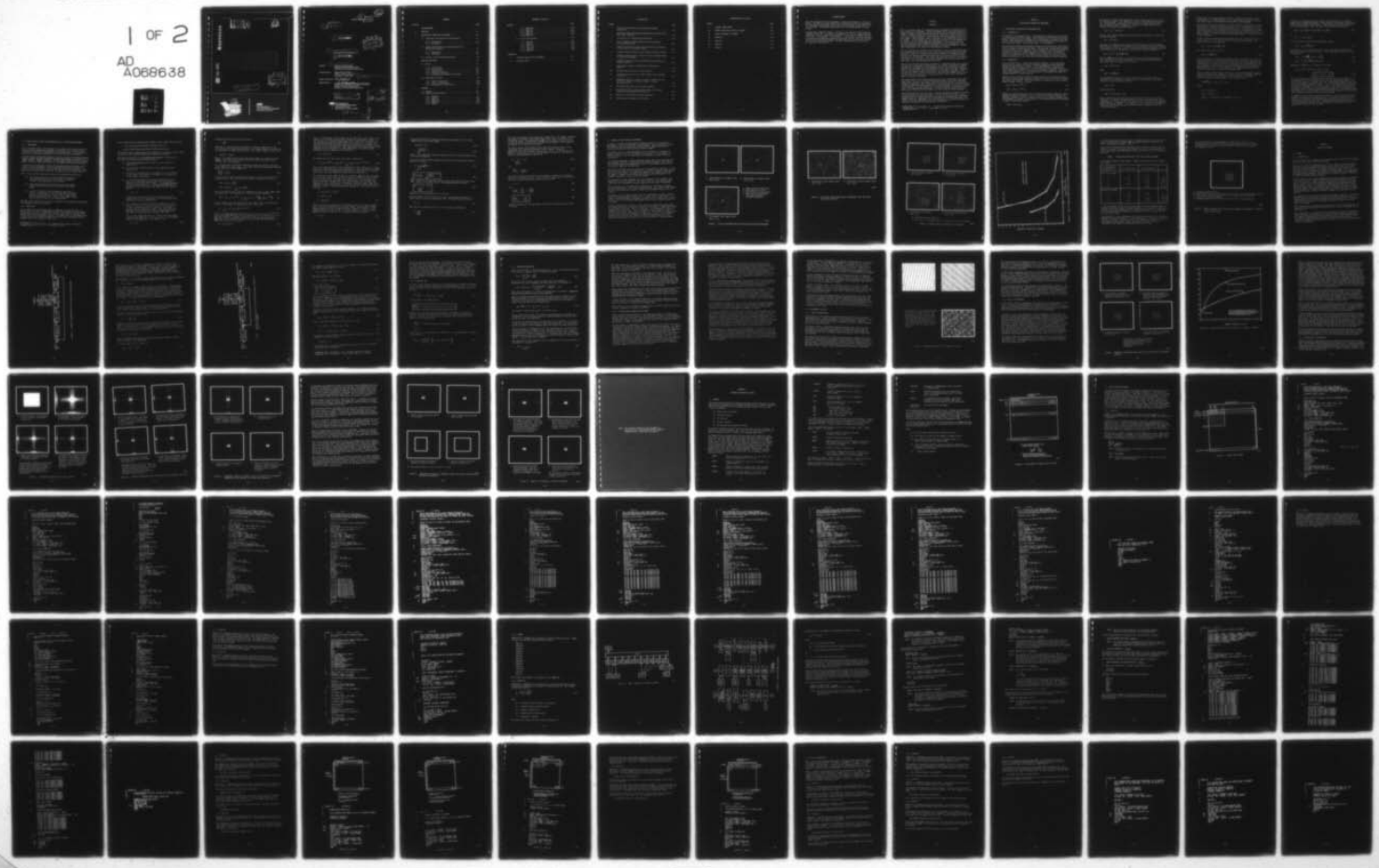
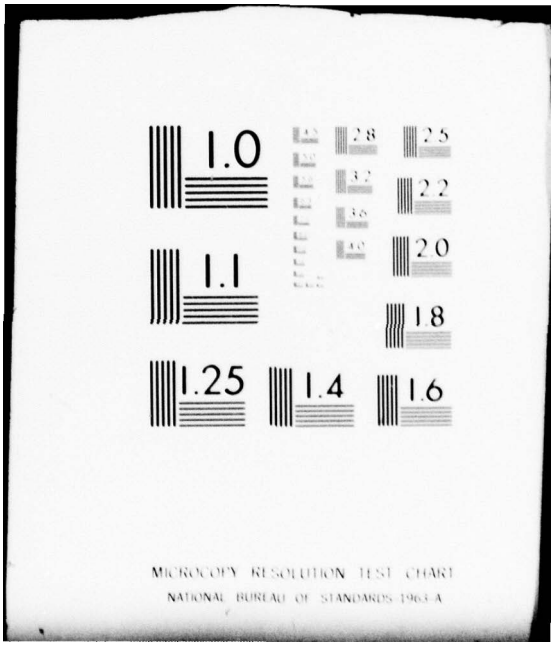
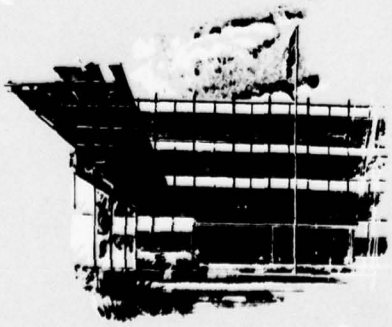
AD-A068 638 FORD AEROSPACE AND COMMUNICATIONS CORP NEWPORT BEACH --ETC F/G 14/5
MACHINE HOLOGRAPHY.(U)
MAR 79 C L RICHARDS N00014-78-C-0659
UNCLASSIFIED U-6516 NL

1 OF 2
AD A068638




1.0
1.1
1.25
1.4
1.6
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.8
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A



Ford Aerospace & Communications Corporation
Aeronutronic Division

(12)

(14) Report Number U-6516

(12) 121 p.

(11) 30 MAR 1979

(6) MACHINE HOLOGRAPHY •

(9) FINAL REPORT

(10) Author: Chester L. Richards
Digital Systems Department
Advanced Development Operation

Prepared for: Department of Navy
Office of Naval Research
Pasadena, California

(15) Under Contract: N00014-78-C-0659

Approved by: C. Richards

Chester L. Richards, Program Manager
Digital Systems Department
Advanced Development Operation

Ford Aerospace & Communications Corporation
Aeronutronic Division
Newport Beach, California 92663

391 853

79 04 09 145

CONTENTS

CONTENTS (Continued)

ILLUSTRATIONS

ILLUSTRATIONS (Continued)

ACKNOWLEDGMENT

The work reported here is an outgrowth of research performed for a Doctoral Thesis. The subject of that effort was an investigation into the possibility that information processing in the central nervous system follows the rules of holography. The results reported here add to the plausibility of that notion.

I would like to thank Dr. Edwin T. Florance of the Office of Naval Research, Pasadena, California. His faith, encouragement and valuable suggestions were responsible for this contract and research. I also wish to express my gratitude to Mr. David Seppala. It was because of his energy, initiative and skill in developing software and performing experiments that the research produced results far beyond those planned. Software documentation was also part of Mr. Seppala's contribution.

SECTION 1

OVERVIEW

This is the final report of a theoretical and experimental investigation into some properties of holograms. Optical holograms suffer from the existence of complex conjugate terms and nonlinearities which add an unavoidable amount of clutter. Because we were able to perform our experiments using a digital computer (Data General Eclipse), these conjugate terms and nonlinearities were not present. This provided a certain cleanness in the mathematics which will be evident. It also showed that the essential properties of the hologram are not restricted to a wave propagation environment. Thus, we have chosen to provide the general name of Machine Holography for our investigation. The name implies that the principles investigated here may be embodied by diverse types of machinery, not just laser based optical systems.

The investigation was divided into two packages. The first explored the ultimate storage capacity of what we have come to call Conventionally Multiplexed Holograms. These are holograms which are multiplexed by a summation process whereby each reference pattern participating is uncorrelated with the other references in the ensemble. This subject was partially explored by La Macchia and White* for certain types of optical holograms, and has been generalized here, while also treating important new cases.

Theoretical treatment shows that the amount of information which may be stored in this type of hologram is proportional to the number of degrees of freedom available in the output of the holographic system. The theory also predicts that it does not matter how the information is divided for storage. There may be only a few pages of data with many bits on each page; conversely, a few bits may be stored on each of many pages. Only the total number of bits to be stored enters the calculation. These theoretical results were confirmed by means of computer simulations.

The primary goal of the second package of work was to demonstrate that a hologram could be trained to reliably distinguish between two highly correlated (reference) patterns. This effort broke entirely new ground. In an effort to design a training paradigm to demonstrate this prediction, a new kind of machine was discovered. This new machine put a negative feedback loop around the hologram. The hologram then became part of a servo mechanism in which both the controller of the servo and the object of control was the stored hologram.

Theory predicted that this new type of adaptive hologram is capable of cleaning up clutter and false reconstructions. It also indicated that this type of hologram could be trained for differential pattern recognition by a remarkably simple procedure. Experiments have confirmed both of these predictions. We report, then, the pioneering of new and potentially very fertile territory for scientific exploration and technological exploitation.

*La Macchia, J. T., and White, D. L. "Coded Multiple Exposure Holograms," Applied Optics, Vol. 7, No. 1, (Jan 1968) pp. 91-94.

SECTION 2

MULTIPLEXING THEORY AND EXPERIMENT

2.1 HOLOGRAPHIC RECORDING AND RECONSTRUCTION

2.1.1 INTRODUCTION

The formation and reconstruction of a hologram can be regarded as an exercise in encoding and decoding an image. The encoding activity involves the linear spatial transformation of the image and an appropriately configured reference pattern. The transformed reference acts as an encoding key and is mixed with the transformed image. The recorded result is the hologram.

Reconstruction involves inserting the reference into the system, transforming it and playing it against the hologram. The reference then decodes the information contained in the hologram. After an appropriate inverse transformation, the image is restored to something like its original form. The degree of similarity between the original image and its reconstruction will be determined by the properties of the reference pattern used, by the nature of the linear transformation, and by the mode of multiplexing which may be employed.

2.1.2 DERIVATION

Let us consider first a simple unmultiplexed hologram. For the following derivations we assume, for clarity, continuum images and transformations. Perhaps the most typical and useful linear transformation is the Fourier transform, which is adopted here. All functions will be annotated with a single independent variable to avoid undue notational complexity. Generalization to two parameters for a full description of two-dimensional images is a straightforward but tedious exercise.

Suppose we have a subject image $s(x)$ which we wish to record and reconstruct. This will be paired with a reference image $r(x)$, which will be used to encode and decode the hologram. These two patterns are connected with the hologram domain through a Fourier transform

$$R(\xi) = \int r(x)\ e^{-i\xi x}\ dx \tag{1}$$

$$S(\xi) = \int s(x)\ e^{-i\xi x}\ dx \tag{2}$$

During the recording process both the reference and subject images are transformed according to Equations 1 and 2. Before they are mixed together the reference undergoes a further transformation into its complex conjugate form. The hologram is recorded as the product of the transformed subject and complex conjugate transformed reference:

$$H(\xi) = S(\xi)\ R^*(\xi) \tag{3}$$

Reconstruction involves the reinsertion of the reference pattern, r(x), into the reference channel. The reference is Fourier transformed and delivered to the hologram domain. During the reconstruction the transformed reference does not undergo complex conjugation. Instead, it simply multiplies the hologram resulting in a response pattern:

$$A(\xi) = S(\xi)\ R^{*}(\xi)\ R(\xi) \tag{4}$$

Finally, this response undergoes an inverse Fourier transform to produce the reconstructed image

$$a(x) = F^{-1}[A(\xi)] \tag{5}$$

The degree of resemblance between the reconstruction a(x) and the original image s(x) will depend on the properties of the reference r(x) under a Fourier transformation. This can be seen by noting that the inverse Fourier transform of the triple product of Equation 4 results in the triple convolution:

$$a(x) = s(x) * \Big(r(x) \circledast r(x)\Big) \tag{6}$$

Where $r(x) \circledast r(x)$ represents an autocorrelation of the reference pattern. This autocorrelation serves the purpose of a system point spread function. The system output can therefore be written

$$a(x) = s(x) * p(x) \tag{7}$$

where

$$p(x) = r(x) \circledast r(x)$$

It is within this context that the significance of the proper choice of reference can be understood. Suppose that the reconstruction point spread function is delta function-like; then

$$p(x) = \delta(x) \tag{8}$$

but by definition

$$a(x) = \delta(x) * s(x) = s(x)$$

Thus, a delta-like reference autocorrelation will produce a perfect reconstruction. In general, we might expect a reference to be delta-like only in approximation. In such a case, the reference acts as a filter which degrades the reconstruction somewhat. For example, if the reference autocorrelation is finite in width it will act like a low pass filter to smooth or blur the reconstructed image.

Another type of reference pattern, which is of particular interest in this investigation, is a random function. The autocorrelation of such a function will usually show a very sharp and narrow spike at autoalignment.

This spike is delta-like and leads to a high resolution reconstruction of the subject image. When the reference pattern is out of autoalignment, the autocorrelation will produce a low level random "noise." The convolution of this noise-like pattern with the subject will result in a scrambled haze of clutter. Thus, holographically associating a subject image with a random reference pattern leads to a sharp reconstructed image superimposed on a low level random clutter background.

Consider next what happens when the reference used for reconstruction, $r'(x)$, is not identical to the reference used for recording the hologram. The reconstruction has, then the form

$$a(x) = s(x) * \Big(r(x) \circledast r'(x) \Big) \tag{9}$$

The system point spread function

$$p(x) = r(x) \circledast r'(x) \tag{10}$$

will be degraded by the degree of mismatch between $r(x)$ and $r'(x)$. The correlation coefficient, taken between $r(x)$ and $r'(x)$, provides a measure of the similarity between these two functions and, therefore, of the quality of the reconstruction. If $r(x)$ and $r'(x)$ are completely uncorrelated the point function of Equation 10 will usually produce a random pattern of low amplitude. This random function, in turn, means that the attempted reconstruction of Equation 9 will contain only low level random noise.

This is a useful property which can be put to use in multiplexing holograms. Suppose we establish an ensemble of reference patterns: $\{r_k(x)\}$. The condition to be met by these patterns is that each member is completely uncorrelated with the other members of the ensemble. Thus

$$r_k(x) \circledast r_L(x) = \delta_{kL}\, \delta(x) + \rho_{kL}(x) \tag{11}$$

where

$\delta_{kL} = 0$, when $k \neq L$

$\delta_{kL} = 1$, when $k = L$

$\rho_{kL}(x)$ = a random function dependent on k and L

If the set of references has this property, multiplexing a set of subject images can be achieved by addition. Each subject is paired with one reference drawn from the ensemble of references. The hologram is formed by adding up all of these pairings. Thus, the form of the multiplexed hologram is:

$$H(\xi) = S_1(\xi)\, R_1^*(\xi) + S_2(\xi)\, R_2^*(\xi) + S_3\, R_3^* + \ldots \tag{12}$$

or

$$H(\xi) = \Sigma_k\, S_k(\xi)\, R_k^*(\xi)$$

If we write this in the output image domain we have

$$h(x) = \Sigma_k\, s_k(x) \circledast r_k(x) \tag{13}$$

Reconstruction of one particular subject will occur when we reinsert the associated reference into the system. In the image domain this process may be written as:

$$a_L(x) = r_L(x) \circledast \left(\Sigma_k\, r_k(x) * s_k(x)\right) \tag{14}$$

$$a_L(x) = \Sigma_k \left(r_L(x) \circledast r_k(x)\right) * s_k(x) \tag{15}$$

Making use of Equation 11 in Equation 15 we have:

$$a_L(x) = \delta(x) * s_L(x) + \Sigma_k\, \rho_{kL}(x) * s_k(x)$$

$$\boxed{a_L(x) = s_L(x) + \text{noise}} \tag{16}$$

The reconstruction of the subject associated with the chosen reference is accompanied by clutter noise generated by various random cross terms. As we increase the number of holograms stored in a multiplexed ensemble we will also get a corresponding increase in noise. This noise ultimately sets a limit on the amount of information which may be stored. This limit will be calculated in detail in Paragraph 2.2. Note the important point that multiplexing by this means (called conventional multiplexing) depends on a correlation process. This correlation is the consequence of recording the hologram in the form of a sum of Fourier transforms of the contributing images.

In summary, the process of holography depends on a convolutional association between a reference and a subject pattern. Reconstruction results from the creation of a synthetic point spread function resulting from the auto- or crosscorrelation of two references. Multiplexing is possible because the use of an uncorrelated reference in the reconstruction results merely in the generation of a haze of clutter noise.

2.2 SIGNAL-TO-NOISE RATIO FOR RECONSTRUCTION OF A MULTIPLEXED HOLOGRAM

2.2.1 BACKGROUND

One of the major tasks of the contract was to explore the storage capacity of a conventionally multiplexed hologram. A prime goal was to find out which factors were significant in determining the storage limit. We also wished to know what the functional dependence was on these significant parameters.

A conventionally multiplexed hologram is defined as a multiply exposed hologram formed by adding together separately recorded subholograms. The result is a summed ensemble. Random encoding of the individual references paired with each subject is used to make each subject in the ensemble selectively retrievable.

A theoretical investigation of this task had previously been carried out for a type of optical hologram*. But this investigation did not consider the possibility that the subjects would be made up of multiple data points. Some questions also existed as to the effect of sampling on the storage capacity.

These issues have now been settled with the following principal results:

(a) The storage capacity of a random reference multiplexed hologram depends directly on the number of degrees of freedom of the image output of the system.

(b) The storage capacity varies inversely with the energy signal-to-noise ratio (clutter) required at the system output.

(c) It does not matter in what configuration the data is stored: a few frames of data with many data points will produce the same signal-to-clutter ratio as many frames with few data points each. Only the total number of data points extracted from all the frames matters.

The last result appears to be new and is important in assessing the performance of holographic data storage systems.

2.2.2 DERIVATION

Let us assume that the hologram plane is composed of n sample points. We wish to store a total of H subholograms within this domain. Each subhologram is presumed to have been formed with a unique random reference pattern, each having a narrow autocorrelation function. We suppose that at each sample point in the hologram the contributed amplitude from any given subhologram in the ensemble has, on the average, a value of α.

---

*La Macchia, J. T., and White, D. L. "Coded Multiple Exposure Holograms," Applied Optics, Vol. 7, No. 1, (Jan 1968) pp. 91-94.

We will find that the reconstruction includes clutter terms from two sources:

- o Cross terms within the proper reference function.
- o Cross terms between uncorrelated reference functions.

The limits of data storage capacity will depend on the amount of clutter which creeps into the reconstruction process from these two sources.

The proper perspective for performing signal magnitude calculations is developed by considering the following points:

(a) Every sampled image may be treated as a vector array of sample values.

(b) The vector has a length derived from an appropriate sum of these values.

(c) A measure of the magnitude of the image vector is the square of its length. This measure corresponds to the intensity of a conventional optical wave.

(d) If we examine the image in the output domain we find that each of its information bearing points is isolated and unaffected by its associates. Thus, if the average amplitude of each of these points is σ, the square magnitude for, p, points will just be the sum of the squares of the individual points:

$$M_I = p\,\sigma^2 \tag{17}$$

Another way of saying this is that the magnitude of the output image vector is determined by a root sum square (RSS) addition of the output image points.

(e) If we examine the discrete Fourier transform (DFT) of this image, we find that the sample values in this new space are no longer independent and do not combine in RSS fashion. In fact, they are coherently linked. This means that their amplitude values add directly, and the square magnitude is the square of the sum of the individual amplitudes.

Thus, if the average amplitude of a transformed image sample in the Fourier domain is α, and if there are, n, such samples, then the square magnitude of the image will be

$$M_I = (n\alpha)^2 \tag{18}$$

Combining Equations 18 and 19 we find that:

$$\sigma = \frac{n\alpha}{\sqrt{p}} \tag{19}$$

The effect of clutter may be calculated by a detailed examination of the reference function during reconstruction. We recognize that the reference may be considered as a sample data function, in the Fourier domain, of the form:

$$\text{reference} = R_k(\xi)$$

Where ξ is an index which locates a particular sample in a space of n such samples. The index k labels the particular reference function from the ensemble, which is being used.

In a conventionally multiplexed hologram the resulting reference involves a linear summation of many terms. If we ignore the associated subject functions we may write a reference ensemble as:

$$\left\{R^*\right\} = \Sigma_k \; R_k^*(\xi) \tag{20}$$

Let us now go through the reconstruction process by inserting a particular reference $R_J(\xi)$. The result will be to output a function of the form

$$P(\xi) = R_J(\xi) \; \Sigma_k \; R_k^*(\xi) \tag{21}$$

$$P(\xi) = |R_J(\xi)|^2 + \sum_{k \neq J} R_J(\xi) \; R_k^*(\xi)$$

Let us next express these reference components in terms of their image domain representations. The connection is a discrete Fourier transform.

$$P(\xi) = \left|\Sigma_x \; r_J(x) \; e^{-i\xi x}\right|^2 + \sum_{k \neq J} \left(\Sigma_x \; r_J(x) \; e^{-i\xi x}\right)\left(\Sigma_y \; r_k^*(y) \; e^{+i\xi y}\right) \tag{22}$$

Let us consider first the second term on the right, which contributes pure clutter. Rearranging the summations in this term we get:

$$C_1 = \sum_{k \neq J} \Sigma_x \; \Sigma_y \; r_J(x) \; r_k^*(y) \; e^{-i\xi(x-y)} \tag{23}$$

This is a random summation evidenced by the fact that the factor $e^{-i\xi(x-y)}$ is nonzero and ranges uniformly over all values from -1 to +1 and from -i to +i. Thus the terms must be summed together RSS to get the equivalent vector magnitude. Let us assume first that the average amplitude independent of phase effects will be α such that

$$\alpha = \langle r_J(x) \; r_k^*(y) \rangle \tag{24}$$

Next, it is necessary to count terms. We note that both x and y range over n samples. So the double sum over these parameters contributes n^2 terms. If the ensemble has H separate subholograms, H-1 of these will contribute to C_1 and the remaining one will result in the reconstruction of the image. We therefore have a total of $(H-1)n^2$ amplitudes contributing to clutter from this component of the reconstruction. Since these terms are randomly phased, their magnitude will be the result of an RSS combination. The clutter square magnitude from this component will therefore be

$$C_1^2 = (H-1)n^2 \alpha^2 \tag{25}$$

We examine next the first term on the right in Equation 22:

$$\left|\Sigma_x r_J(x)\ e^{-i\xi x}\right|^2 = \Sigma_x \left|r_J(x)\right|^2 + \sum_{x \neq y} \Sigma r_J(x)\ r_J(y)\ e^{-i\xi(x-y)} \tag{26}$$

The first term on the right is the reconstruction term. There are n of these terms each represented by an average amplitude α. Note that there is no phase variation so that the summation of amplitudes (over the index x) is coherent. This leads to the square magnitude result given by Equation 18.

The second term on the right involves an incoherent summation and contributes to the clutter. (Note, that for certain types of reference functions, such as a delta function in the image plane, this term is zero and results in a perfect reconstruction when there is no multiplexing.) We recognize that there must be n^2-n contributions of amplitude α from this term. Since the summation of these contributions is incoherent they will be RSS added and will therefore combine in square magnitude with Equation 25. The total clutter is therefore

$$C^2 = (H-1)n^2\alpha^2 + (n^2-n)\alpha^2$$

$$C^2 = Hn^2\alpha^2 - n\alpha^2$$

$$C^2 = (Hn-1)n\alpha^2 \tag{27}$$

This is the clutter as measured in the Fourier domain. Its impact, however, will be on the reconstruction in the image domain. Suppose that the image domain contains a total of m sample points or degrees of freedom. If the average amplitude of the clutter in the image domain is ν, then the square magnitude of this clutter will be (assuming incoherent summation)

$$C^2 = m\nu^2 \tag{28}$$

By equating Equations 27 and 28 we determine the average clutter at each sample point of the output domain:

$$n(Hn-1)\alpha^2 = m\nu^2 \tag{29}$$

$$\nu = \sqrt{\frac{n}{m}(Hn-1)}\ \alpha \tag{30}$$

(Note: The assumptions leading to Equations 29 and 19 involved implicit use of Parseval's theorem.)

Now we can estimate the mean signal-to-clutter ratio by taking the ratios of Equations 19 and 30.

$$\frac{\sigma}{\nu} = \frac{n\alpha}{\sqrt{\frac{pn}{m}(Hn-1)}\ \alpha} \tag{31}$$

$$\boxed{\left(\frac{\sigma}{\nu}\right) \equiv \left(\frac{s}{N}\right)_A = \sqrt{\frac{nm}{P(Hn-1)}}} \tag{32}$$

$\left(\frac{s}{N}\right)_A$ represents an amplitude signal-to-noise ratio where the noise is due solely to crosstalk clutter. Rearranging Equation 32 we get

$$\boxed{PH = \frac{m}{\left(\frac{s}{N}\right)_A^2} + \frac{P}{n}} \tag{33}$$

Equation 33 is the primary result of interest.

The total amount of data stored in the type of multiplexed hologram is P points stored in each of H holograms or PH. We define the data total as:

$$PH \equiv D \tag{34}$$

Note that for a sufficient ratio of n to p we have, approximately:

$$D \approx \frac{m}{\left(\frac{s}{N}\right)_A^2} \tag{35}$$

Thus, the total amount of data stored is in proportion to the number of samples (or degrees of freedom) in the OUTPUT image field. This is a new and very interesting result. An important consequence of Equations 34 and 35 is that it does not matter how the division is made between numbers of data pages (H) and numbers of data points (P) per page.

The inverse square relationship with the amplitude signal-to-clutter ratio corresponds to an inverse relationship to the "energy" (intensity) signal-to-clutter ratio familiar from coherent optics. In the special case where the output image plane has the same number of degrees of freedom as the Fourier or hologram plane, we may write:

$$D = \frac{n}{\left(\frac{s}{N}\right)_A^2} + \frac{P}{n} \tag{36}$$

and

$$\left(\frac{s}{N}\right)_A = \frac{n}{\sqrt{PHn-P}} \tag{37}$$

An example calculation is useful in establishing a baseline for the experiments to be described shortly. Suppose we have an image field of 4096 points and that our data field has many fewer points, so that

$$P << n \tag{38}$$

then

$$\left(\frac{s}{N}\right)_A = \frac{\sqrt{n}}{\sqrt{PH}} = \frac{\sqrt{4096}}{\sqrt{PH}} \tag{39}$$

$$\boxed{\left(\frac{s}{N}\right)_A = \frac{64}{\sqrt{PH}}} \tag{40}$$

Equation 40 describes the experimental conditions and results which we wish to verify.

2.3 RESULTS OF MULTIPLEXING EXPERIMENTS

In order to validate the theoretical arguments presented in the previous paragraphs, a series of experiments was performed. The experiments involved the formation and reconstruction of multiplexed holograms using random reference patterns and conventionalized subject images.

The primary object of these experiments was to determine the dependence of signal-to-clutter ratio on the amount of data stored. Also, we wished to find out whether or not the data limit was independent of page formatting, as predicted.

In a typical experiment, several reference subject pairs were associated and added together to form a conventionally multiplexed hologram. Then each of the reference images was submitted, in turn, to the hologram and the associated subject image was reconstructed.

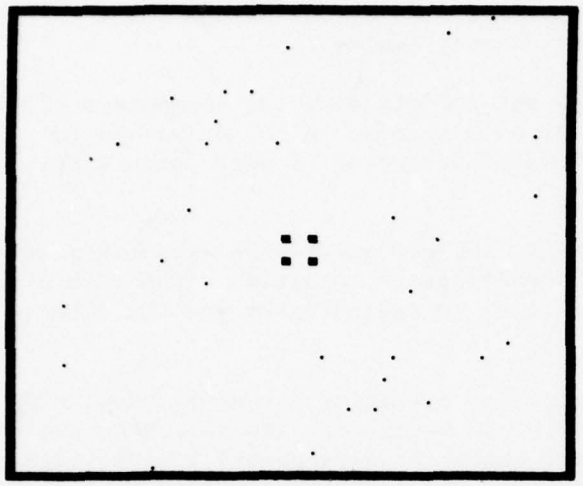
The signal-to-clutter ratio was determined by averaging the amplitudes, of the data points and comparing this value with an average of samples taken from the data-free border region. By varying the number of data points in the image, the number of samples in the field, and the number of subholograms in an ensemble a parametric study could be performed.

The apparatus for this experiment was a Data General Eclipse computer, combined with a Printronix printer. The print characters were under software control and were programed to produce images with ten levels of grayscale.

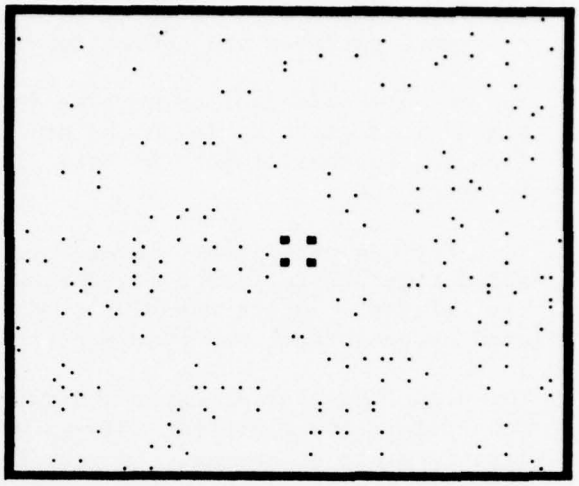
The image format for the experiments presented here consisted of a square array of 4096 samples. The Fourier transform plane also contained 4096 samples. The data arrays were also chosen as square arrays ranging from one point to sixteen.

In Figures 1 and 2, sample outputs from five experiments are shown. The data image for these experiments consisted of a square array of four points. In order to show that proper independent reconstruction was taking place the data arrays used to form different subholograms were located in different parts of the scene.

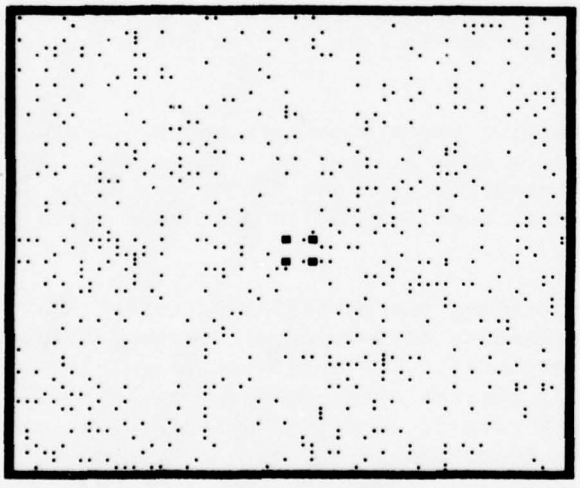
A cursory examination of Figures 1 and 2 shows the qualitative feature that the signal-to-clutter ratio (S/N) decreases as the number of images multiplexed in an ensemble increases. This is as expected. The quantitative results conformed to the theory, also. This is illustrated by the numbers of Table I, and the plot of these numbers given in Figure 4. The theoretical values given in the second column are computed using Equations 34 and 35. The standard deviation given in the last column is compiled from measurements of the reconstruction signal-to-clutter ratios of each member of a given ensemble.



A. Nonmultiplexed, one image stored
S/N = 33.56



B. Multiplexed, two images stored
S/N = 21.54

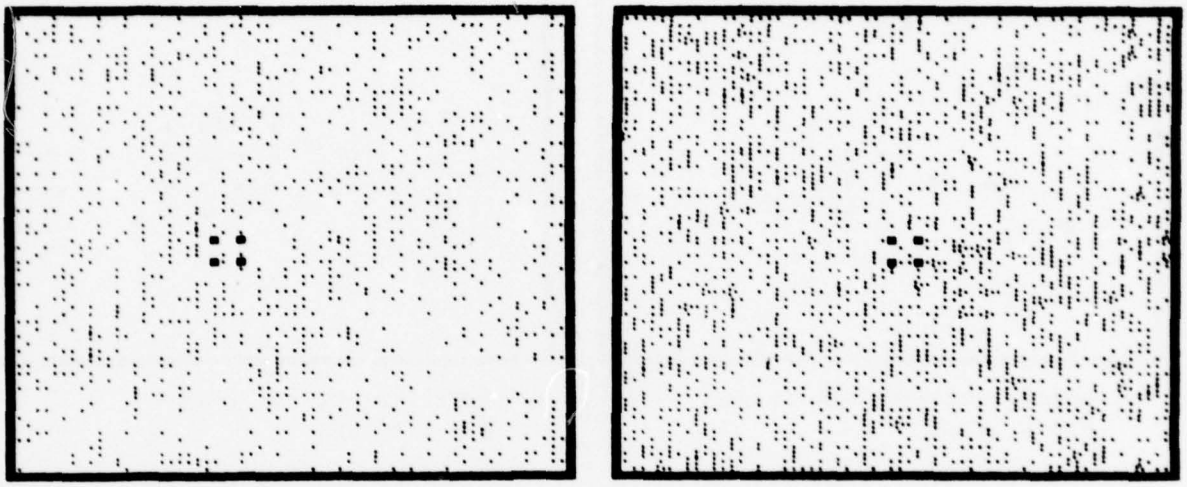


C. Multiplexed, four images stored
S/N = 15.71

- Sample field has 4096 points
- Data Image has 4 points
- S/N ratios are for particular images being reconstructed
- Each image represents a different experiment

93-8-17

FIGURE 1. TYPICAL RECONSTRUCTIONS OF MULTIPLEXED HOLOGRAM EXPERIMENTS

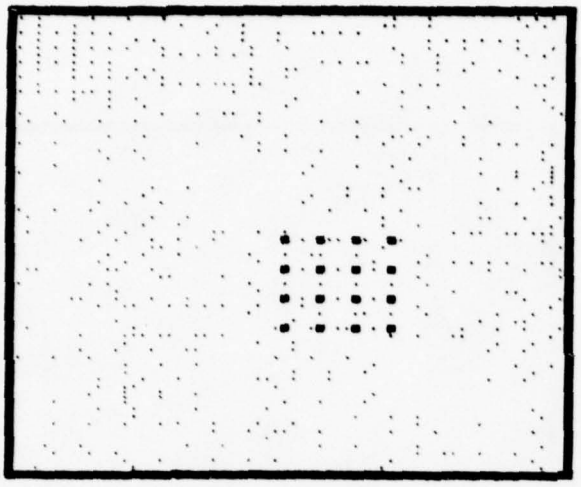


A. Multiplexed, eight images stored
S/N = 10.97

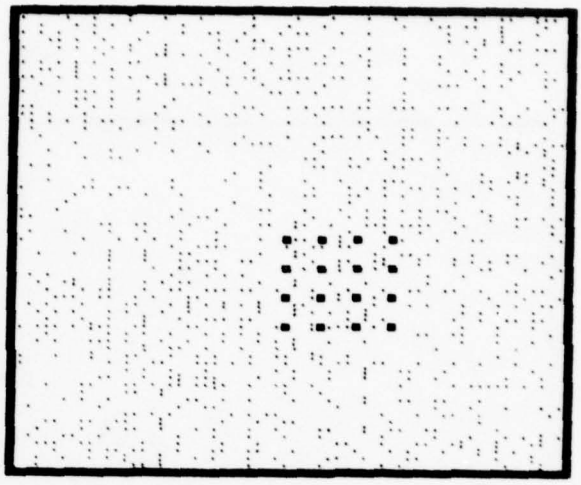
B. Multiplexed, sixteen images stored
S/N = 8.023

93-8-18

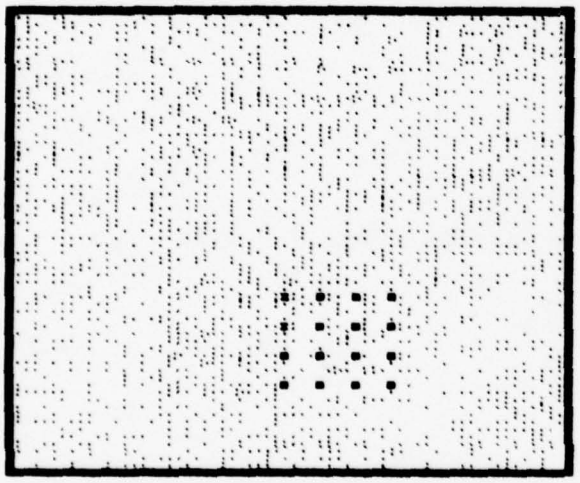
FIGURE 2. ADDITIONAL SAMPLE MULTIPLEXING EXPERIMENTS WITH FOUR POINT DATA IMAGES (SUBJECT)



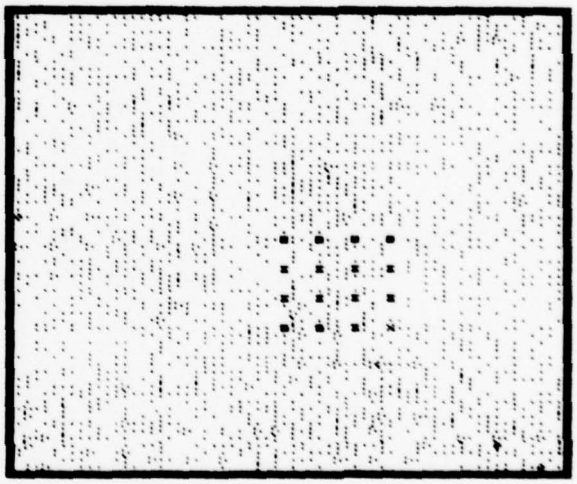
A. Nonmultiplexed, one image
S/N = 14.96



B. Multiplexed, two images
S/N = 11.38



C. Multiplexed, four images
S/N = 8.22



D. Multiplexed, eight images
S/N = 5.99

- Sixteen-point data image used
- Each image is a representative reconstruction from one experiment

93-8-19

FIGURE 3. SECOND SERIES OF MULTIPLEXING EXPERIMENTS

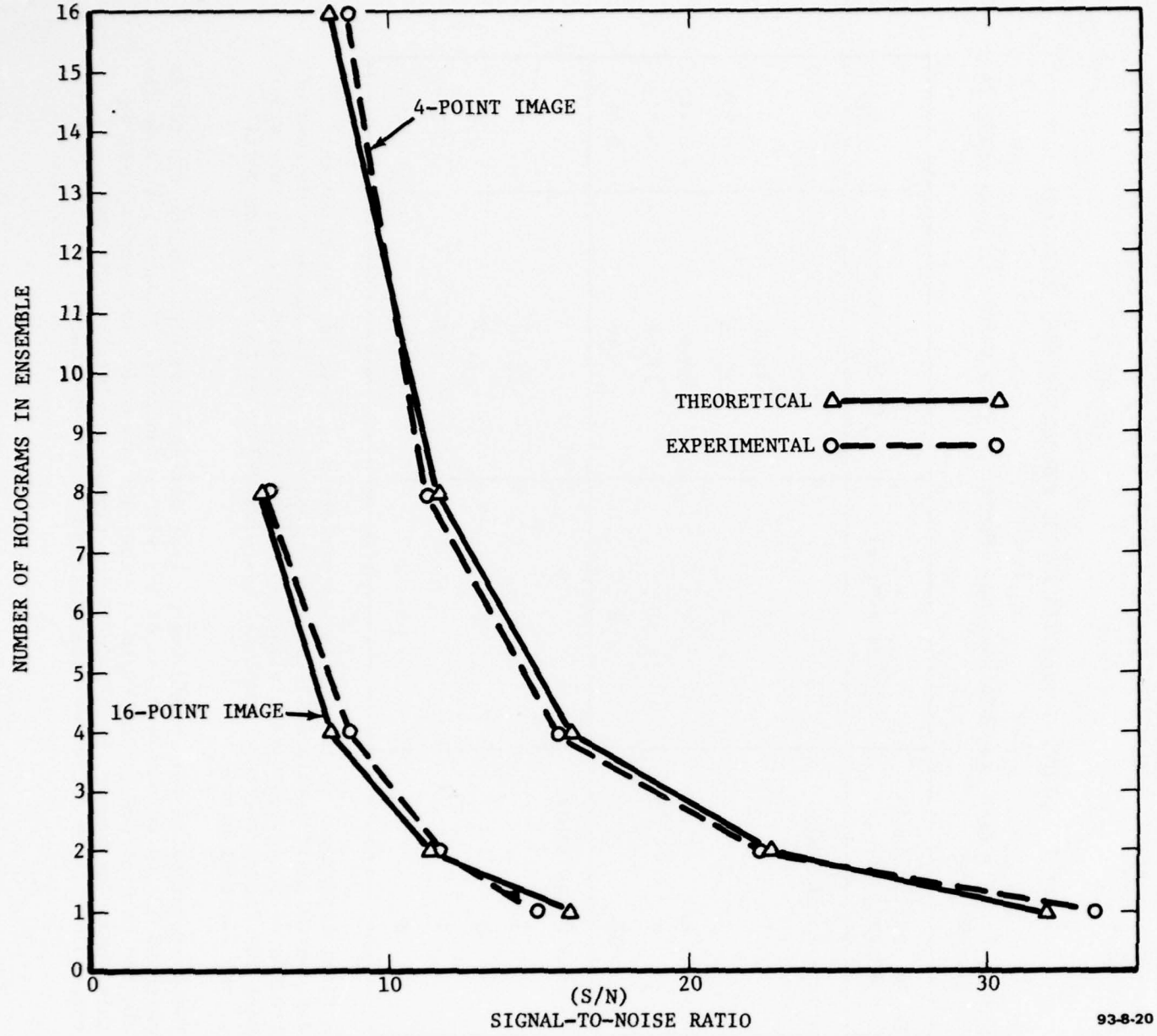


FIGURE 4. PLOT OF SIGNAL-TO-CLUTTER RATIOS FOR CONVENTIONALLY MULTIPLEXED HOLOGRAMS

It should also be noted that the use of a random reference results in some net clutter even when only a single image is being recorded and reconstructed. This is illustrated by Figure 1, view A.

This series of experiments was repeated for the case where the data image consisted of an array of sixteen points (Figure 3). Again, as demonstrated in Table I and Figure 4 the experiments and the theory are in strict quantitative agreement.

TABLE I. SIGNAL-TO-CLUTTER RATIOS FOR MULTIPLEXED HOLOGRAMS

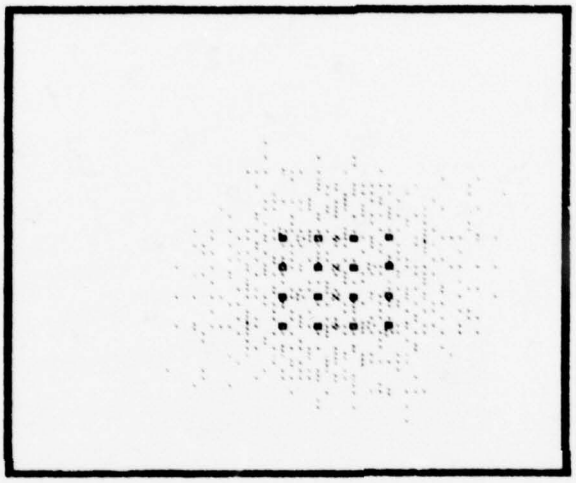
Output image domain has 4096 samples. Number of data points in each image is as listed.

No. of Multiplexed Holograms (H)	S/N Theoretical	S/N Experimental	σ_n
4 Data Point Image			
1	32	33.56	-
2	22.63	22.36	0.825
4	16	15.66	0.916
8	11.31	11.45	0.313
16	8	8.66	0.662
16 Data Point Image			
1	16	14.96	-
2	11.31	12.27	0.89
4	8	8.676	0.57999
8	5.65	5.674	0.196

A final experiment in this class was performed to test the assertion that the data which could be stored is primarily dependent on the number of degrees of freedom of the output image plane, in accord with Equation 35. In this experiment the random reference employed was windowed so as to be a minor portion of the field of view.

The consequence of this windowing is that during reconstruction the clutter is weighted by the autocorrelation of the window function. Moreover, because the clutter is confined in the field of view, its magnitude is enhanced. It is this enhancing factor which reduces the signal-to-clutter ratio and, consequently, the amount of information which can be stored.

The reconstruction of this experiment is illustrated in Figure 5. The weighting by the autocorrelation of the window function is clearly evident. The enhancement of clutter is evident when this illustration is compared with Figure 3, view A.



- Hologram is unmultiplexed
- Clutter field tapered according to auto correlation of reference window function
- Peak clutter very much higher than if it were more widely distributed (compare with Figure 3, View A
- No attempt made to measure signal-to-clutter ratio in this experiment

93-8-21

FIGURE 5. SAMPLE RECONSTRUCTION WITH ORIGINAL REFERENCE CONSTRAINED TO A FRACTION OF THE FIELD OF VIEW

SECTION 3

ADAPTIVE HOLOGRAMS

3.1 THEORY

3.1.1 INTRODUCTION

It was asserted by the proposal that a hologram could be trained to distinguish between two very similar reference patterns.

The notion was that the common subset of the two reference patterns could be canceled, leaving distinguishing features which could be used to reconstruct identifying subject images. The technique makes use of the ability of a hologram to coherently subtract by means of a phase change. The isolation of the characteristic features was considered to be analogous to the Gram-Schmidt orthogonalization technique. This concept is explored in detail in Appendix A, which is borrowed from the proposal.

During a design of the training algorithm, which would implement this concept, it was realized that a negative feedback process would achieve the desired goal. The idea was to wrap a feedback link around the outside of the holographic system so that the reconstructed image would be subtracted from the current subject input. The holographic recording mechanism would use this difference image, combine it with the current reference, and add the resulting subhologram to the preexisting ensemble according to the rules of conventional multiplexing. Figure 6 provides a schematic representation of this process.

By supplying two different paired subject and reference patterns to the system inputs according to some sort of alternation scheme, it was felt that the common subsets of the reference patterns would be subtracted off. This would leave the reconstructions to the mutually othorgonal features of the two reference patterns.

This configuration was extremely interesting, but it involved a great deal of computation in performing the forward and inverse Fourier transformations for each iteration.

After some study it was realized that, because of the linearity properties of these transforms, the feedback loop could be brought inside the transforms. The adjustment process then would be entirely in the Fourier domain. Except for initial transformations to prepare the subject and reference and an inversion to display the ultimate result, there would be no need for repetitive Fourier transformations.

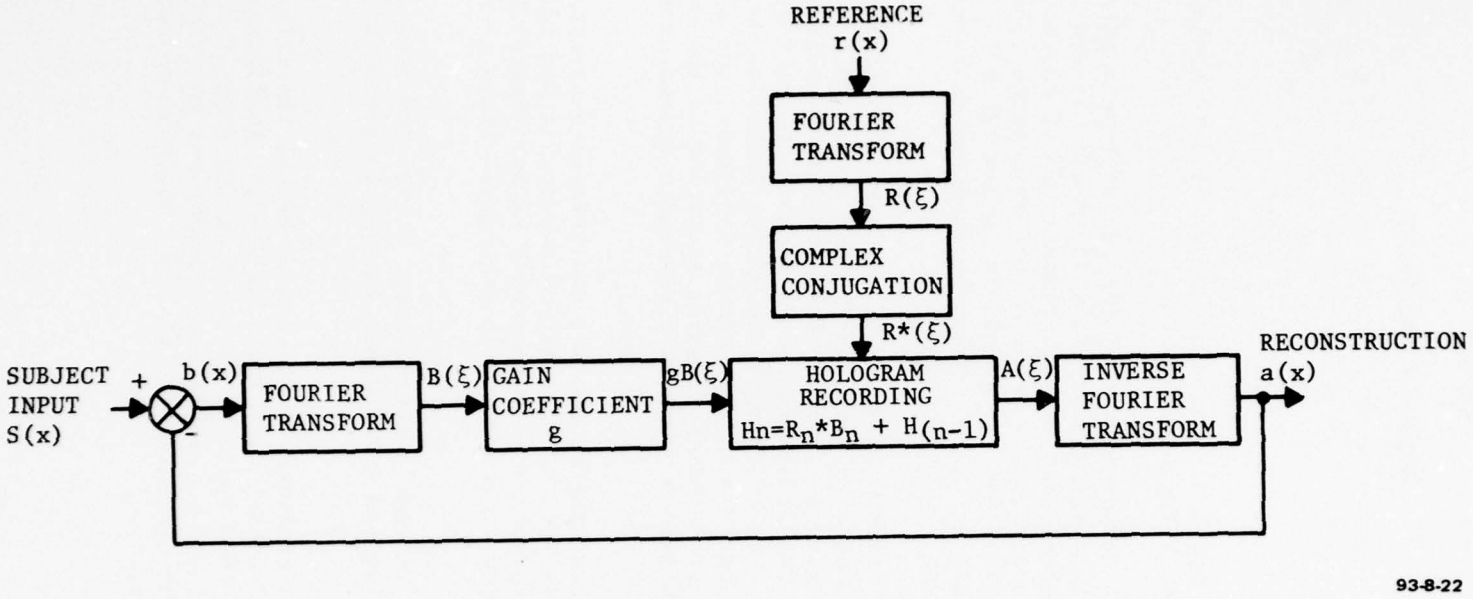


FIGURE 6. SCHEMATIC REPRESENTATION OF THE ORIGINAL TRAINING SCHEME

This change led to an adaptive holographic system which is schematically illustrated by Figure 7. This system is easy to analyze and has a number of exceptionally interesting properties. Implementation of the arrangement in software showed that it can be unstable. Some minor adjustments in the description of the iteration process insured unconditional stability without violating the clarity of the basic idea. These matters will be discussed in detail in the next section.

The possibility of this simple feedback arrangement, with its powerful properties, was a surprise. It is the outstanding result of this study effort.

3.1.2 BASIC THEORY

In the adaptive holographic scheme the set of values, $H_n(\xi)$, of the hologram, at any given time, is derived from a summation of past inputs. The process is related to conventional holographic multiplexing because of the formation of an ensemble of subholograms. At each step of the process the current patterns arriving through the subject and reference channels are multiplied together. This product forms a new subhologram which is added to the preexisting ensemble of holograms. The equation which describes this summation is

$$H_n(\xi) = gB_n(\xi)\, R_n^*(\xi) + H_{n-1}(\xi) \tag{41}$$

In this equation $H_{n-1}(\xi)$ is the preexisting multiplexed hologram. $H_n(\xi)$ is the new multiplexed hologram. $R_n(\xi)$ is the current reference pattern, $B_n(\xi)$ is the current subject pattern and g is a loop gain constant.

The subject pattern is derived from two sources. One is a Fourier transformed version of the current subject input

$$S_n(\xi) = F\left[s_n(x)\right] \tag{42}$$

Subtracted from this is a pattern $A(\xi)$. $A(\xi)$ is the consequence of an instantaneous reconstruction process which is on-going. This process is only implicit in Figures 6 and 7. The conjugate reference, $R^*(\xi)$, is used to form the hologram, as shown. The unconjugated reference $R(\xi)$ is used to reconstruct the hologram according to the procedure

$$A_n(\xi) = R_n(\xi)\, H_n(\xi) \tag{43}$$

$A_n(\xi)$ is inverse transformed, upon demand, and thereby produces an image, $a_n(x)$, appropriate for examination.

By subtracting A_n from S_n we get B_n.

$$B_n(\xi) = S_n(\xi) - A_n(\xi) \tag{44}$$

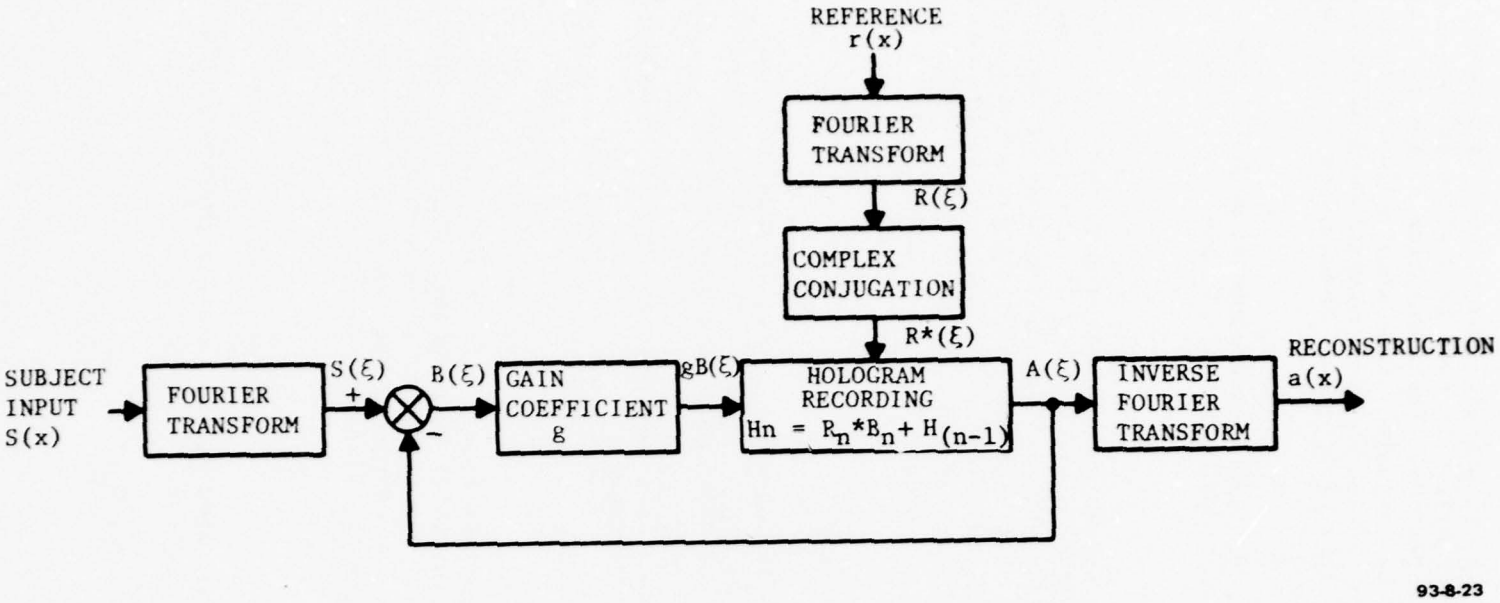


FIGURE 7. SCHEMATIC REPRESENTATION OF MOST EFFICIENT TRAINING SCHEME FOR ADAPTIVE HOLOGRAMS

The combination of Equations 41, 43, and 44 results in a difference equation. For example, using Equations 41 and 44

$$H_n = g\,(S_n - A_n)R_n^* + H_{n-1} \tag{45}$$

Employing Equation 43 in 45 gives

$$H_n = g\,S_n R_n^* - g\,|R_n|^2\,H_n + H_{n-1} \tag{46}$$

We can solve for H_n

$$\boxed{H_n = \frac{H_{n-1} + g\,S_n R_n^*}{1 + g|R_n|^2}} \tag{47}$$

This is the proper iterative difference equation to describe the operation of the adaptive hologram. It is unconditionally stable. Notice that in the derivation of this equation it was assumed that the formation of the increment to the hologram ensemble, $g(S_n - A_n)R_n^*$, used a reconstruction, A_n, which itself depended on the increment. This seems like a bootstrap operation and is counter intuitive. The procedure is known as backward differencing and is essential for stability.*

Suppose we were to use a more intuitive approach. Let us make the subject channel input depend on the reconstruction of what has already been recorded. Thus

$$H_n = g\,R_n^*\,S_n - g\,R_n^*\,A_{n-1} + H_{n-1} \tag{48}$$

Now, we substitue Equation 43 into Equation 48 to get

$$H_n = g\,R_n^*\,S_n - g\,R_n^*\,R_{n-1}\,H_{n-1} + H_{n-1} \tag{49}$$

$$H_n = (1 - g\,R_n^*\,R_{n-1})H_{n-1} + g\,R_n^*\,S_n \tag{50}$$

Equation 50 is the forward difference equation version of the process. It is unstable for positive g when

$$g\,R_n^*\,R_{n-1} \geq 2. \tag{51}$$

For negative g it is always unstable, which is also true for the backward difference case of Equation 47.

*Oppenheim, A.V., and Schafer, R.W., "Digital Signal Processing," Englewood Cliffs, New Jersey: Prentice-Hall (1975), pp. 203-206.

In the initial series of experiments on the adaptive hologram the forward difference Equation 50 was employed. We found that the process would converge for awhile, but then would begin to rapidly diverage. We were using random reference patterns for these experiments. In order to get the convergence time within reasonable limits, the value of g was made moderately large. It would turn out that some spatial frequency component of the power spectrum of the reference, $|R_n(\xi)|^2$, would be large enough to satisfy Equation 51. This spatial frequency would exhibit positive feedback and the system would blow up. By switching to the backward difference Equation 47, we completely cured the problem.

3.1.3 CONTINUUM CASE

In order to gain further insight into the properties of the adaptive hologram, let us carry the differencing process to the infinitesimal limit. If the step interval between n-1 and n represents a time Δt, we may rearrange Equation 46 and write:

$$\frac{H_n - H_{n-1}}{\Delta t} = -g\,|R_n|^2\,H_n + g\,S_n\,R_n^* \tag{52}$$

Carrying this to the limit we get:

$$\boxed{\frac{dH(\xi,t)}{dt} = -g\,|R_n(\xi,t)|^2\,H(\xi,t) + g\,S(\xi,t)\,R^*(\xi,t)} \tag{53}$$

Equation 53 is the most general description of the adaptive holographic process. With specialization of the terms we can draw out some interesting results. We choose as a special case a reference and subject which both switch on to a constant level.

$$\frac{dH(\xi,t)}{dt} = -g\,|R(\xi)|^2\,H(\xi,t) + g\,S(\xi)\,R^*(\xi) \tag{54}$$

for $t>0$

If $H = 0$ for $t \leq 0$, the solution of Equation 54 is straightforward, and results in the equation:

$$H(\xi,t) = \frac{S(\xi)\,R^*(\xi)}{|R(\xi)|^2}\left[1 - \exp\left(-g\,|R(\xi)|^2\,t\right)\right] \tag{55}$$

3.1.4 IMPORTANT PROPERTIES

There are two things to note about Equation 55. First, the hologram converges, in the limit of infinite time, to the spatial pattern.

$$H(\xi) = \frac{S(\xi)\ R^{*}(\xi)}{|R(\xi)|^{2}} = \frac{S(\xi)}{R(\xi)} \tag{56}$$

We see that the reference pattern is thrown into the denominator. Reconstruction of the hologram, using this reference, is described by:

$$A(\xi) = R(\xi)\ H(\xi) = \frac{S(\xi)\ R^{*}(\xi)\ R(\xi)}{|R(\xi)|^{2}} = \frac{S(\xi)\ R(\xi)}{R(\xi)} = S(\xi) \tag{57}$$

Thus, the reconstruction is a perfect reproduction of the subject, <u>regardless of the function which is used for the reference</u>.

This is a substantial improvement over the conventional recording and reconstruction process. Even if a completely random function is used for the reference, there will be no background haze with this type of hologram. In terms of the description give by Equation 11,

$$r_{K}(x) \circledast r_{L}(x) = \delta_{KL}\ \delta(x) + \rho_{KL}(x), \tag{11}$$

the random crosstalk term, $\rho_{KL}(x)$, is forced to zero.

We can regard this outcome as a sidelobe cancellation process whereby the sidelobes of the point spread function, produced through the autocorrelation of the reference, are suppressed.

(It has not escaped notice that the functional form of Equation 56 is essentially the same as that employed in image restoration work. It may very well be that a variant of this technique will prove useful for image restoration.)

The second issue of consequence arises from consideration of the functional ratio of Equation 56. Selection of an arbitrary function, $r(x)$, to serve as a reference will usually result in the transformed pattern, $R(\xi)$, taking on the value zero for certain frequencies, ξ. We might expect that the hologram, apparently having zeros in the denominator, will be infinite in value at some points. In fact, this does not happen.

The resolution of this problem is found by examination of the time constant of the exponent in Equation 55:

$$\tau(\xi) = \frac{1}{g|R(\xi)|^{2}} \tag{58}$$

The time constant is a function of spatial frequency which is inversely proportional to the power spectral density. For those spatial frequencies at which $R(\xi) = 0$, the time constant is infinite and no change in the hologram takes place during the adaptation.

There are interesting correlaries of this important result. First, all portions of the hologram do not adapt at the same rate. Thus, the hologram is not formed as a unit. Second, spatial frequencies where the power spectral density is highest will adapt most quickly and come to equilibrium. For most reference scenes these will usually correspond to the dc background and slowly varying terms. High spatial frequency terms are usually low in amplitude and will adjust slowly. This means that finely detailed structures and features will have influence only after a substantial time has passed.

These results may be summarized by the rule that the basic form and shape (low frequencies) of an object will tend to impact the hologram before the fine details and texture (high frequencies). This rule will be violated only when the high spatial frequencies dominate the power spectrum as when a scene is composed mostly of contrasty textures.

A final correlary is that spatial frequencies of a subject image will not be recorded unless the corresponding spatial frequencies exist in the reference. This rule is true for conventional holography also.

These properties of the adaptive hologram will be of great significance if the concept is employed for pattern recognition.

3.1.5 DISTINGUISHING BETWEEN TWO PATTERNS

Given the interesting properties of the adaptive hologram, it becomes relatively simple to set up a global training paradigm to train it to distinguish between two similar patterns. The patterns to be recognized are fed in through the reference channel. Associated with each reference is a subject pattern which serves to label its associate.

We did a number of experiments in which these patterns were fed to the system in different orders with variable amounts of adaptation permitted. In a typical experiment, a particular reference-subject pair was fed in and many iterative cycles were computed so as to permit the adaptation to go nearly to completion. Then a second reference subject pair would be introduced, again with substantial adaptation. At the end of a double pairing process, a test would be made to see whether or not false reconstructions occurred. This test consisted of applying each reference in turn and seeing whether or not a trace of the unassociated subject appeared. We usually found that it was necessary to flip back and forth between the associated pairs several times before all traces of false reconstruction were rendered invisible (that is, suppressed below the lowest level of the print gray scale).

We also found that the disappearance and form of this crosstalk was path dependent. Thus, it made a difference how much adaptation was permitted before the reference-subject pairs were exchanged. This peculiar result led to some doubts about the nature of the pattern recognition process. It seems that the Gram-Schmidt explanation outlined in the proposal (and Appendix A) may not be correct. In order to verify the validity of the experiment an existance proof was constructed. This proof, given in Appendix B, shows that the adaptive hologram can be trained to completely distinguish between two partially correlated reference patterns.

3.1.5.1 Critical Issue, Superpacking. The generalization of this property of differential recognition to separation between three or more similar patterns has not been demonstrated, as yet. Proof that this is possible for an arbitrary number of patterns would have very important consequences.

Perhaps the most significant consequence would be that the amount of information which could be stored in a hologram would be vastly increased. It was demonstrated in an earlier section that the storage capacity of a conventionally multiplexed hologram is proportional to the number of degrees of freedom of the system in which the hologram is embedded. The limitation was established primarily by the reconstruction crosstalk between various members of the multiplexed ensemble.

If the adaptive apparatus were used to form a multiplexed hologram ensemble it may be possible to completely cancel the crosstalk between ensemble members. If this proves to be the case, then the storage capacity of the hologram increases to the square of the number of degrees of freedom.

The argument behind this conclusion runs as follows: The hologram represents a vector space with n degrees of freedom. We may therefore embed n vectors in this space. Each vector represents a different reference-subject pairing. During the reconstruction the subject vector is reproduced along with a crosstalk term. If the crosstalk can be canceled away, then the subject vector will be in pristine form. If the subject vector also has n degrees of freedom, it therefore can represent this many bits of data. Since, with crosstalk cancellation, there can be n of these subject vectors reconstructed by the system, the total number of data bits which can be stored is $n \times n = n^2$.

This hypothesis, of course, has yet to be validated. If it proves to be correct, it will have at least the following important consequences. First, the amount of effort to pack the hologram full of information will be very great. This is because the crosstalk cancellation process involves an iteration among the contributing reference-subject pairs. Moreover, the more pairs are to be stored, the greater the number of iterations that will probably be required to achieve convergence.

Second, addition of information to an adapted hologram requires a great deal of adjustment. The new image will generate cross terms with all the other images previously stored. These will have to be adapted out one by one. Moreover, this first order adaptation process will in turn create new cross terms which will require at least second order correction. Clearly, the amount of iteration required to stuff a new pair into an ensemble will be a very strong function of the number of old pairs already making up the ensemble.

A third consequence is that the adaptive hologram may provide an immensely powerful mechanism for pattern recognition. This is because, if crosstalk cancellation happens, the number of reference patterns which may be recognized will be equal to the number of patterns which may be embedded. Which is, of course, the number of degrees of freedom.

Consider a practical case. It is within the state of the art to provide Fourier transforms, over an array of 4096 elements, many times a second. If the above argument is correct, a pretrained system of this size could be made to distinguish among as many as 4096 similar and different patterns in a single pass (milliseconds).

The possibility that this superpacking of the hologram may work sets a clear direction for follow-on research. We must discover if full cross talk cancellation is possible or if the rules of conventional multiplexing set the limit. If superpacking proves possible, then an investigation should be made to find an efficient training scheme. Once this basic research work is successfully completed practical applications will rapidly follow because the technology base is ripe.

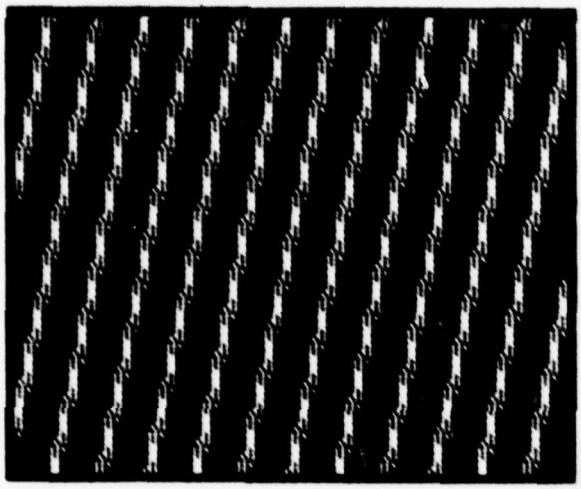
3.2 EXPERIMENTAL RESULTS

3.2.1 TESTS OF ADAPTATION

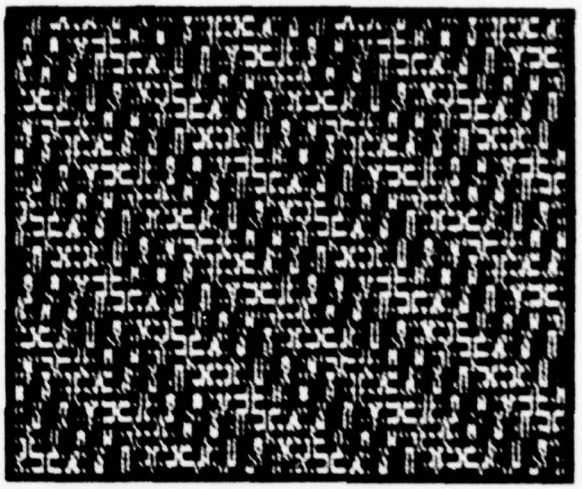
The discovery of the adaptive hologram substantially modified our original experimental plan. We had intended to move straight into an exercise of training a hologram to distinguish between two similar patterns. Instead, we decided to first explore some of the convergence and clutter rejection properties of the adaptive system.

Our initial series of experiments employed a single subject-reference pair. The subject was a data array of sixteen uniform amplitude points, and the associated reference was a random pattern. The choice of these images was made because we had just concluded a detailed properties study of this class of holograms.

After some wrestling with the software and adjusting the feedback loop gain coefficient, we found that the adapting system gradually cleaned up the clutter. This was a notable and exciting event. Further iteration resulted in divergence, however. The reconstructed subject patterns showed characteristic periodicities indicating that the divergence was occurring at localized points in the Fourier plane. Figure 8 shows a representative sequence of divergence.

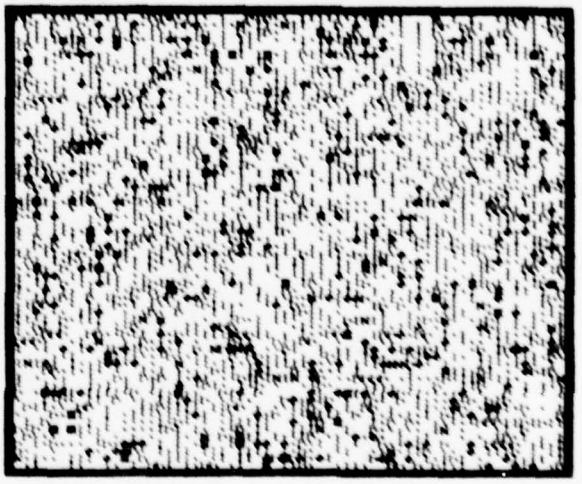


A



B

This sequence of reconstructions shows the effect of runaway feedback when forward differencing is used. Note the periodic structures indicating that high amplitude spatial frequencies begin the runaway process. Lower amplitude frequencies catch up as the high amplitude terms saturate resulting in the last figure in the sequence.



C

938-14

FIGURE 8. RECONSTRUCTIONS EFFECTED BY RUNAWAY FEEDBACK

Upon investigation the problem was traced to our use of a forward difference equation to describe the feedback process. In the preceding theoretical section there is a detailed discussion of this issue. By switching to a backward difference equation, the instability was eliminated and no further problems of this kind were encountered.

A typical early experiment with the corrected system is illustrated by Figures 9 and 10. Figure 9 shows reconstructions of the data field subject with the image showing the characteristic clutter caused by the autocorrelation of the random reference pattern. Note that the initial signal-to-clutter ratio is close to the ensemble average value of sixteen predicted by theory. The other entries in Figure 9 graphically demonstrate the improvement in reconstruction quality as the system iteratively adapts.

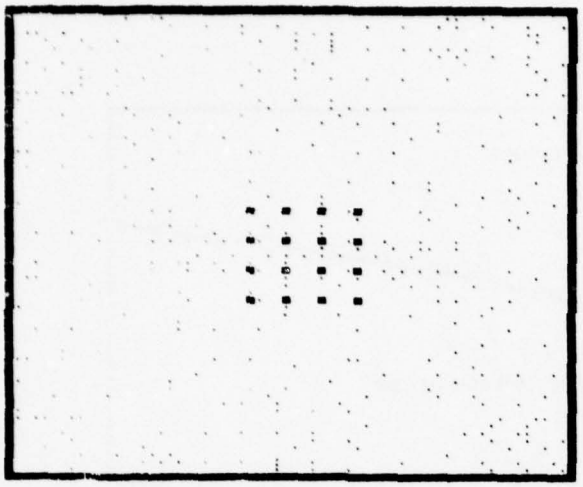
The quantitative results of this experiment are illustrated by Figure 10. The subject amplitude was 100. Initially the reconstructed image had a much lower amplitude. This was a consequence of the values chosen for the magnitudes of the reference pattern, and the loop gain coefficient, g. Upon iteration the amplitude of the reconstruction exponentially converged to that of the subject in accord with Equations 55 and 57. The signal-to-clutter ratio also improved dramatically during this convergence process.

3.2.2 TESTS OF CONVERGENCE

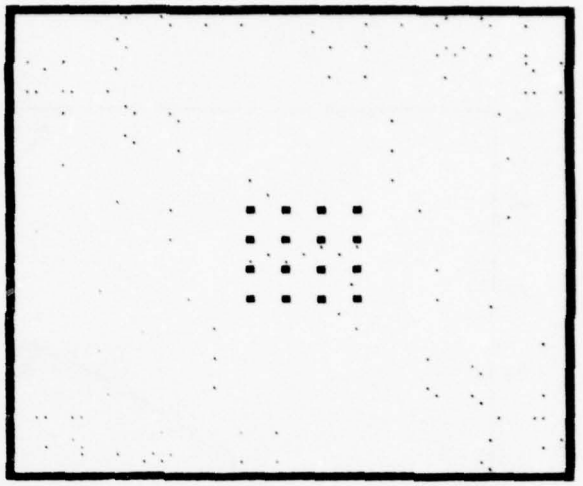
In the initial series of experiments the reference was a random pattern. This type of reference had approximately a uniform power spectral density. The theory, as exemplified by Equation 58, indicated that, in this case all spatial frequencies in the reconstruction would converge at approximately the same rate. This was apparently the case in the initial set of experiments.

We decided to try a much sterner test of the convergence of the system. We selected a reference pattern almost completely dominated by low spatial frequencies with the high frequency terms being relatively very weak. This reference was paired with a subject which was small, complicated and composed primarily of very high spatial frequencies. In conventional holography there would be no chance of getting a useful reconstruction of the subject, but the new theory predicted that, after adaptation, a clear reconstruction was possible. The reference for this experiment was a two dimensional square wave, half filling the field of view. The Fourier transform of this function has the usual form of Sin ξ/ξ and thus assures there is at least some signal at the highest spatial frequencies of the spectrum. This function also contains regions of zero signal, which provides an interesting test of the theory.

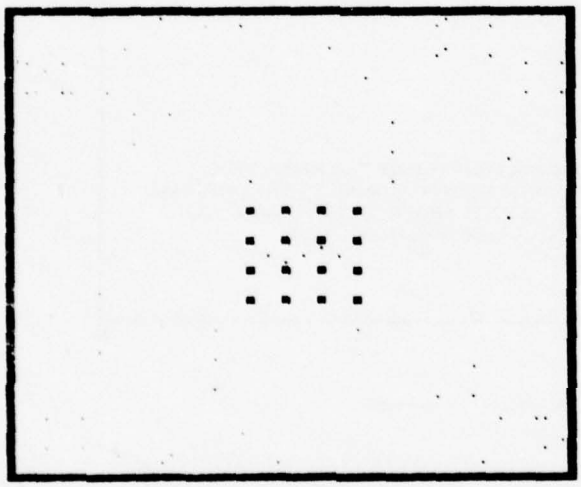
The subject chosen to be associated was the image of the letter character, W (see Figure 14B). The structures of this character were one pixel wide and, therefore, most of the signal energy was concentrated at the high frequency end of the spatial spectrum.



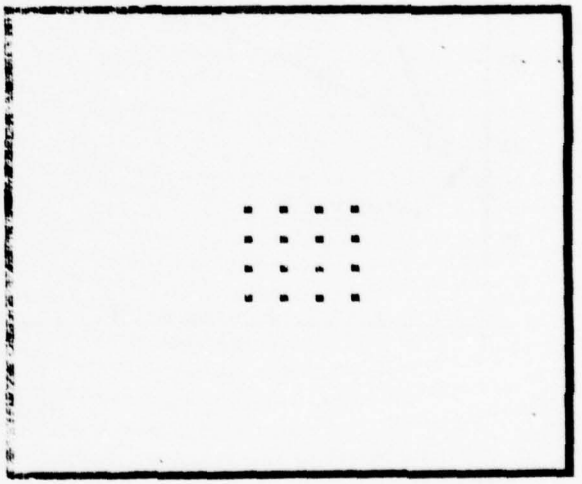
A. Initial frame of sequence. Reconstruction has S/N of 15.11, compared to theoretical of 16.



B. Reconstruction after partial adaption, showing partial clean of autocorrelation produced clutter. S/N = 21.26



C. Continued cleanup of clutter. S/N = 24.30



D. Late reconstruction in adaptive sequence. S/N = 32.59

The clutter is initially generated through the autocorrelation of a random reference image. No multiplexing occurred in this experiment.

9389

FIGURE 9. SEQUENCE OF RECONSTRUCTIONS SHOWING CLUTTER REDUCTION BY FEEDBACK ADAPTION

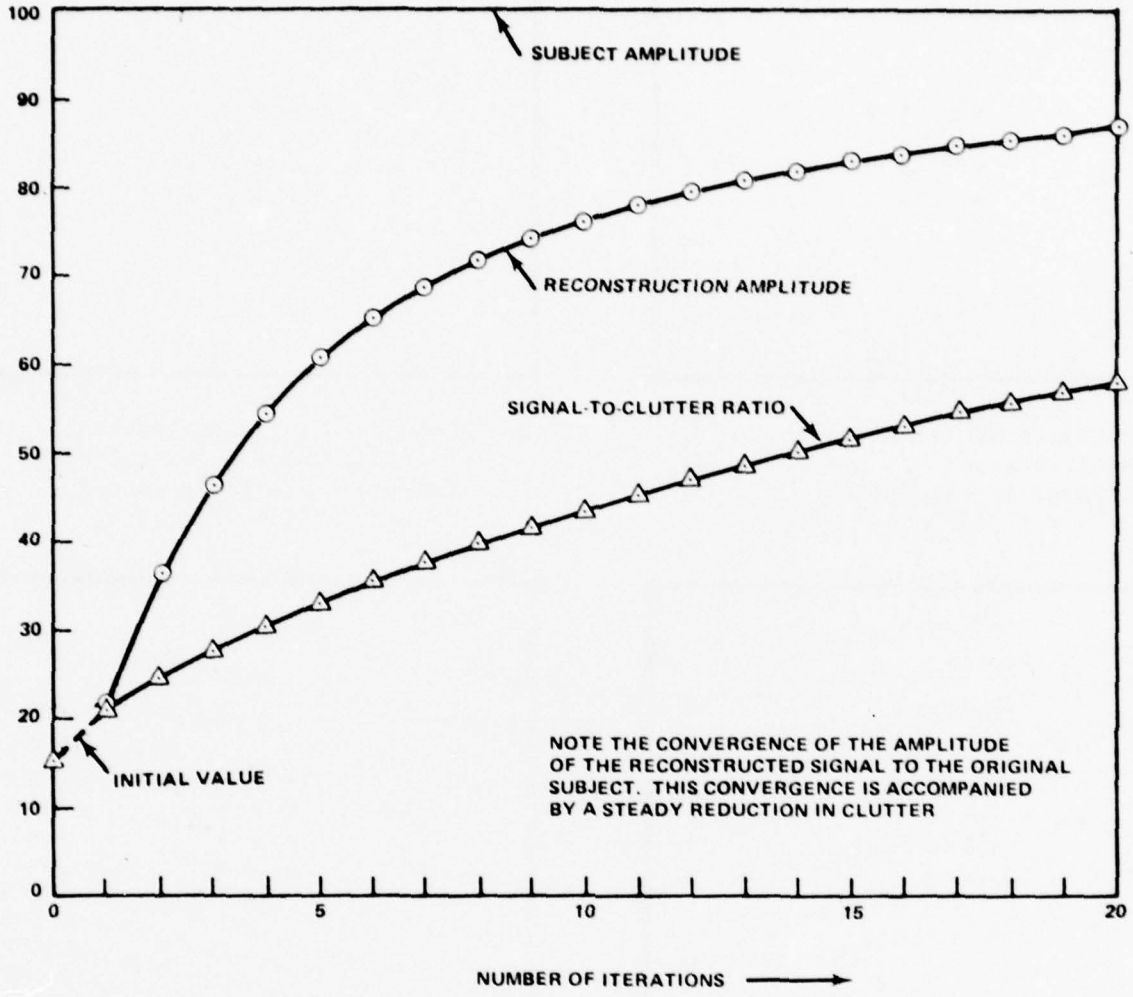


FIGURE 10. QUANTITATIVE RESULTS FOR EXPERIMENT ILLUSTRATED IN FIGURE 9

93-8-10

Figure 11 illustrates the reference used, together with reconstructions after various numbers of iterations. The early reconstructions lack any useable detail, as indicated by Figure 11B. Here the energy of the low spatial frequencies in the reference totally dominate the holographic process. This image is representative of (although somewhat better than) conventional holography. After many iterations the characteristic shape of the W becomes visible. Figure 11D provides an example, late in the iteration sequence of this particular experiment. There are some peculiar features of the process indicated by this reconstruction. Since the effects of the low spatial frequencies have been adapted away, the high frequency terms are dominant. Note the traces of the corners of the reference indicated by the secondary reconstructions at the left and bottom edges of the field. These reconstructions illustrate that cross correlation between the corners is very strong. Note also the influence of the edges of the reference function which is apparent in the streaks radiating from the principal (central) reconstruction.

Since convergence was slow, we repeated the experiment with a much stronger feedback coefficient. The results are shown in Figure 12. The first figure in the sequence apparently picks up at the end of the previous experiment. As the sequence progresses the primary and secondary reconstructions become clear and accurate representations of the original subject. The impact of the edges of the reference function have nearly faded by the end of the sequence, and the reconstruction process is dominated by the corners.

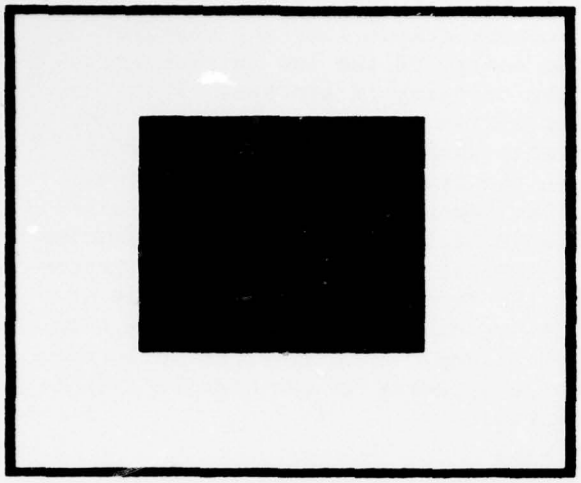
There is some fluctuation in the secondary reconstructions as the convergence process progresses. The left-hand reconstructions start to increase slightly in strength as the bottom reconstruction begins to fade. Presumably, the left reconstructions would peak and fade if the iteration process continued long enough (about two hours of CPU time was used for this experiment).

A third experiment in this series validated the preliminary conclusions drawn from the earlier experiments. For this experiment the square wave was thinned down to just its edge. The reference thus took the form of an empty box (see Figure 14D). This enhancement of the high spatial frequencies in the reference greatly speeded the convergence; Figure 13 evidences the result. In the early iterations the reconstruction looks much like the late forms of Figure 12. By the end of the experiment, the principal reconstruction is nearly perfect, whereas traces of the secondary reconstructions are nearly eliminated.

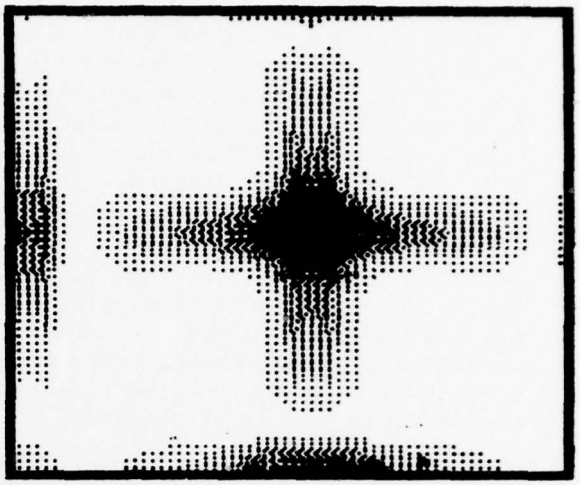
We conclude from these tests that the major features of the theory of adaptation are confirmed. More detailed quantitative studies might show some unexpected phenomena. However, we felt that there would be a greater return for our efforts by pressing on to a test of differential pattern recognition.

3.2.3 RECOGNITION OF DIFFERENCES

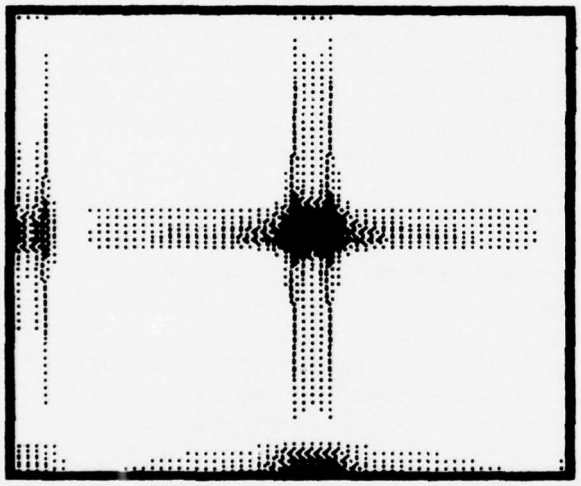
Having successfully demonstrated that the adaptive hologram follows the theoetical predictions, the experimental effort returned to the originally scheduled task. As described in the foregoing theoretical section, it was believed that a hologram could be trained to distinguish between two similar (highly correlated) patterns. With the machinery of the adaptive hologram available, a test of this idea was relatively easy.



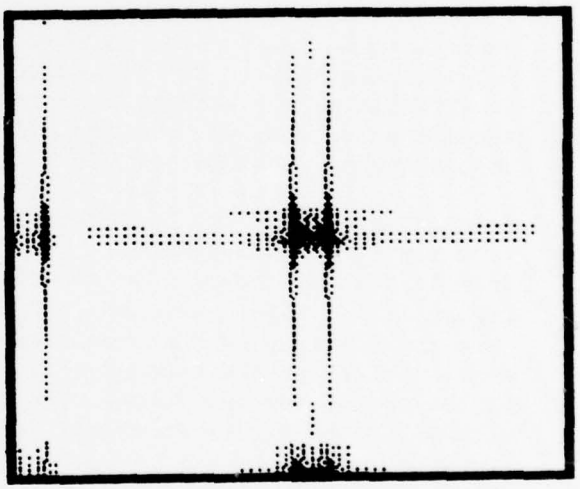
A. Reference pattern. This pattern consists mostly of low spatial frequencies.



B. Reconstruction after first series of iterations. Note dominance of low spatial frequencies.



C. Further in the iteration sequence. High spatial frequency structure is becoming apparent

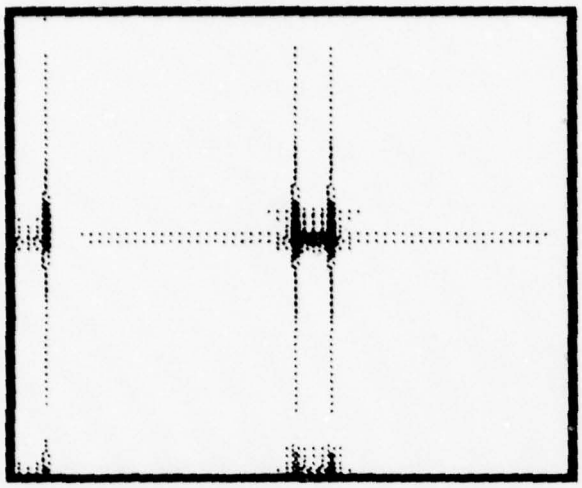


D. Reconstruction at end of test run. The shape of the subject is becoming clear. Reconstruction is now dominated by the reference edge structure. Repeat images are the result of high cross correlation among the corners of reference box image.

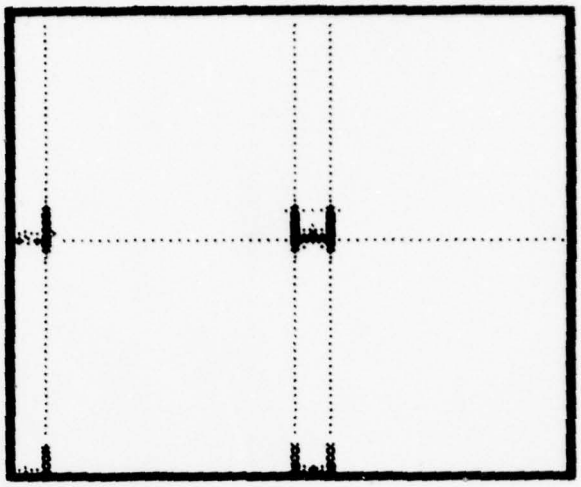
This experiment attempts to reconstruct a small highly structured subject (the image of the letter W) with a much larger nearly unstructured reference. In conventional holography the reconstruction would show virtually no structure.

93-8-11

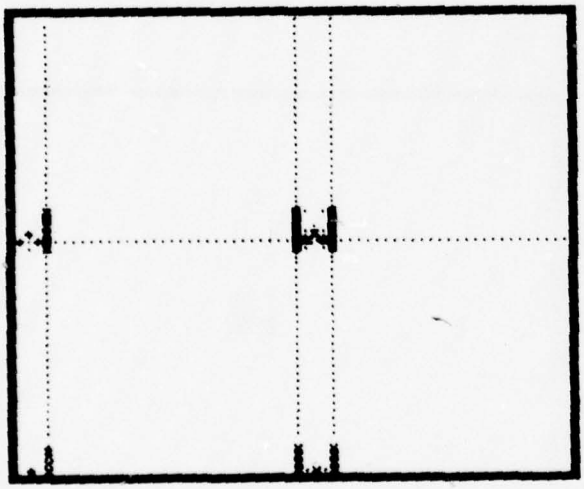
FIGURE 11. ATTEMPTED RECONSTRUCTION OF SMALL SUBJECT



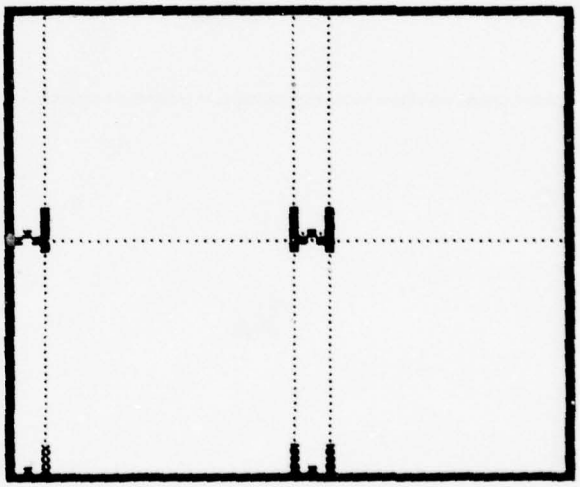
A. Initial reconstruction. The result is very similar to the final result of Figure 11. This experiment is properly regarded as a continuation of that experiment.



B. Reconstruction after 75 iterations. The W structure is becoming clear. Repeat of the image is caused by corner cross correlations.



C. Continued clean up of main image and further weakening of secondary reconstructions.

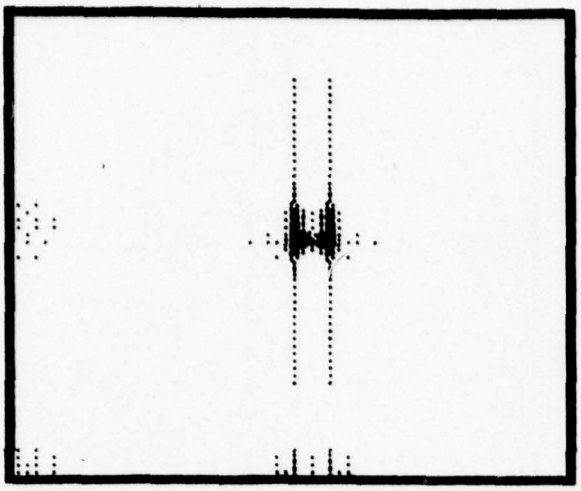


D. Final Image in experiment. Basic trends continue to prevail except for a slight enhancement of the upper left secondary image.

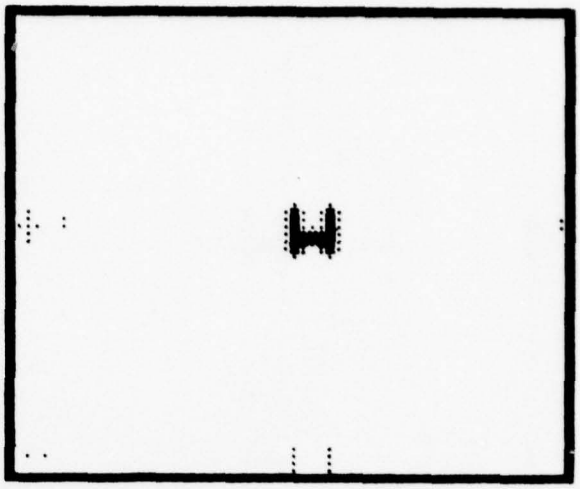
Experiment similar to that of Figure 11, but with increased loop gain. The final reconstruction clearly shows the desired image in the center of the frame. Secondary reconstructions should fade out after further iteration.

938 12

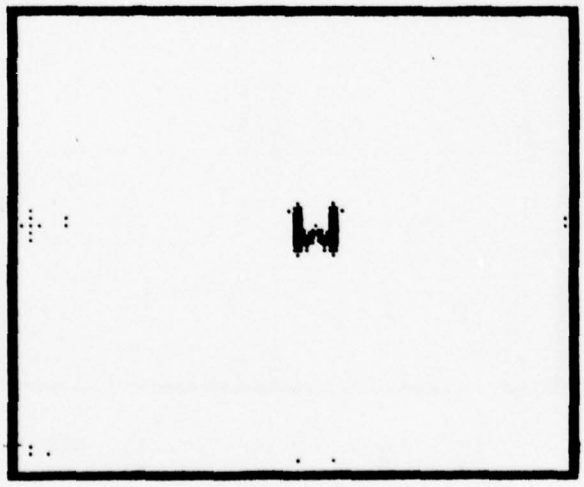
FIGURE 12. ATTEMPTED RECONSTRUCTION OF SMALL SUBJECT WITH INCREASED LOOP GAIN



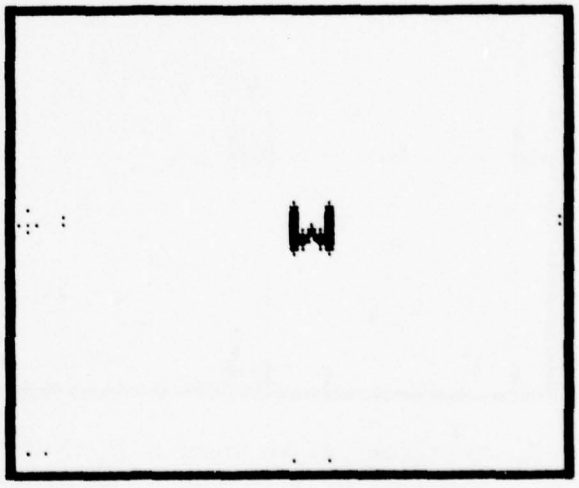
A. Reconstruction at beginning of iteration series. Note Similarity to 12A. Secondary reconstructions start out weaker in this series.



B. Continued clean up of reconstruction.



C. Further reduction of secondary reconstructions



D. Final reconstruction in test sequence. Primary image is almost completely clean. Secondary reconstructions are nearly eliminated.

93813

FIGURE 13. EXPERIMENT SIMILAR TO FIGURES 11 AND 12, EXCEPT THAT THE REFERENCE CONSISTED OF JUST THE EDGE OF THE BOX (SEE FIGURE 14D)

An experiment was designed in which two different subject-reference pairs were presented, alternately, to the system for training. Upon presentation of one such pair, the system was allowed to adapt for several iterations and a test was then made of the reconstructive properties of the partially trained hologram. This consisted of separately presenting the two reference patterns to see what form the reconstructions would take. After this test the alternate reference-subject image pair was injected and the iteration continued.

Typical results are illustrated in Figures 14 and 15. In Figure 14 the reference and subject images are shown. The images were chosen based on previous experience with the system convergence properties, and by the need for simplicity to avoid tedious hand entry of data.

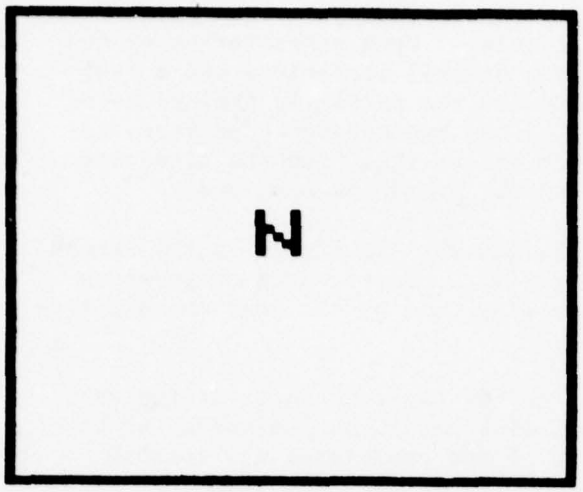
As illustrated a narrow, rectangular box served as the reference for the subject letter image N. A wide, square box was selected as the reference to be paired with the letter image W. The purpose of the experiment was to show that after suitable training, according to the simple paradigm just described, the narrow box would reconstruct only the image of the N, whereas the wide box would reconstruct only the W.

The cross correlation coefficient between the two reference patterns was calculated to be 0.689, so a failure of the system to distinguish between the two reference patterns would show up as strong secondary reconstructions. In order to make these shadow reconstructions evident, the N and W images were located in slightly different positions within the field of view.

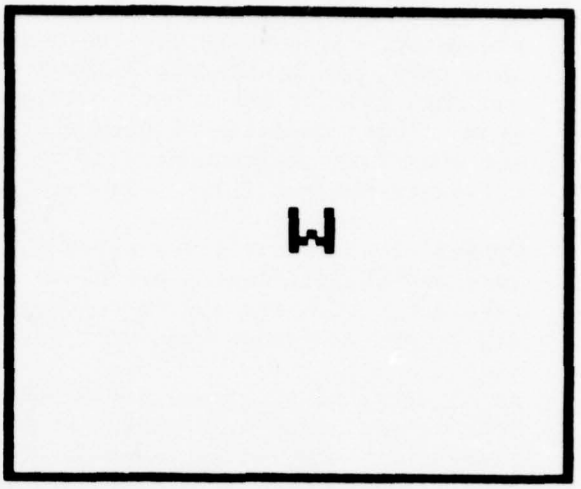
Figure 15 demonstrates the success of this experiment. Figures 15A and 15B show intermediate results after a certain amount of adaptation has occurred. The desired primary reconstructions are becoming clear. Substantial secondary reconstructions of the unpaired subject shows the correlated subset of the two references has not been entirely cancelled out.

The last two subfigures picture the state of the system after further adaptation. The reference subject pair used immediately before this test reconstruction was the W-square box. This shows in the perfectly reconstructed image of the W. It also shows in the slight imperfections of the N image when the narrow box reference is played through the hologram. Although this was the end of the test sequence, convergence was not yet complete. Evidently spatial frequencies of low amplitude were still impacting the reconstruction process and causing false reconstructions.

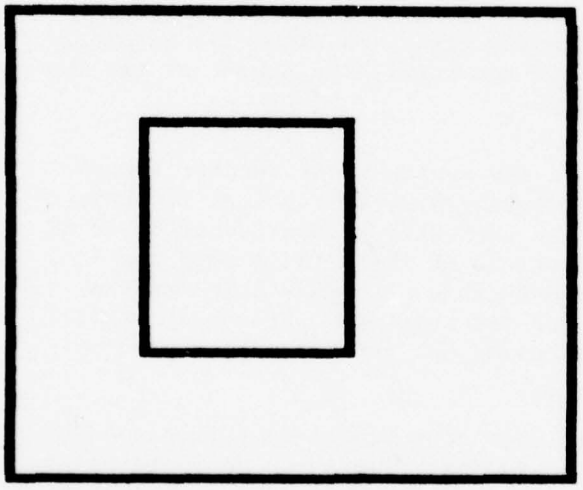
Other experiments of a rough sort indicated that convergence is path dependent. That is, it matters what the strength of the feedback is or how many iterations are made before switching reference-subject pairs. The degree of cross correlation between different reference patterns also appears to have an, as yet, undetermined impact. Both of these topics may be important and should have priority in future investigations.



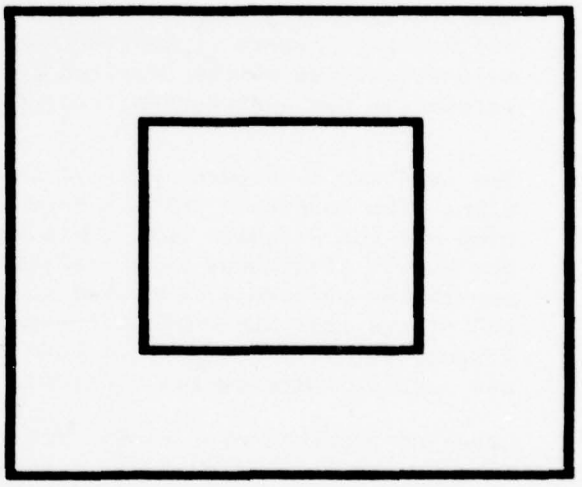
A. Subject image paired with narrow box reference



B. Subject image paired with wide box reference



C. Narrow box serves as reference pattern for subject letter N

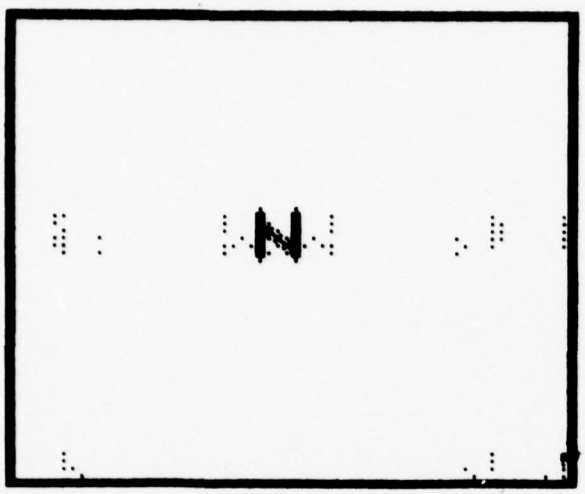


D. Wide box serves as reference pattern for image of W

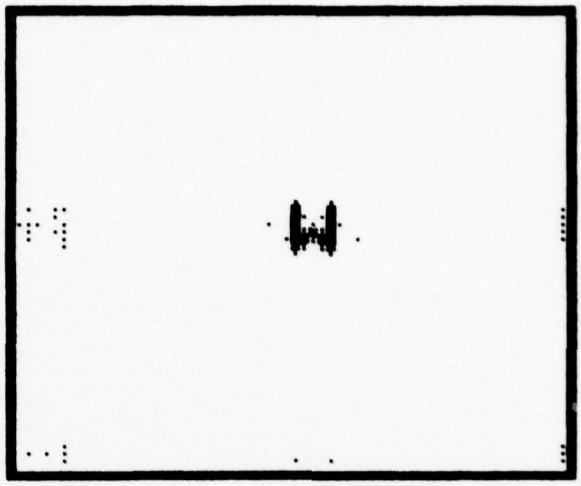
The correlation between the two references is 0.689

93-8-15

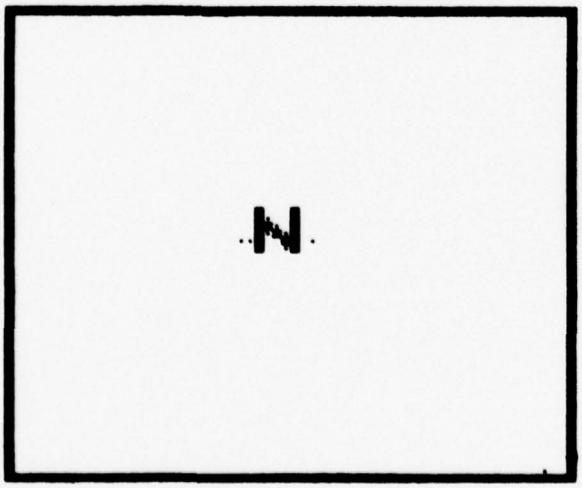
FIGURE 14. CONTRIBUTING SUBJECT AND REFERENCE IMAGES FOR PATTERN SEPARATION AND RECOGNITION EXPERIMENT



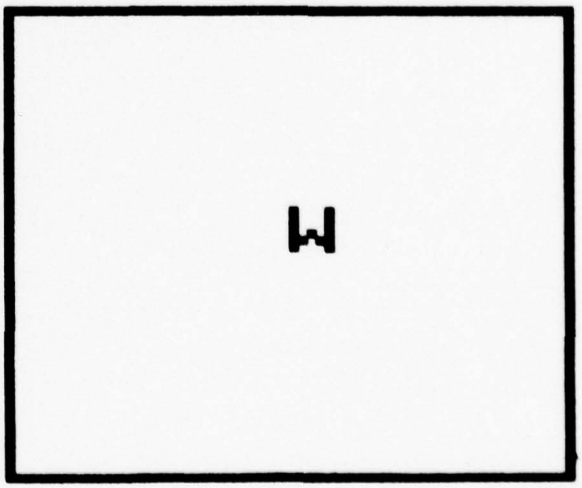
A. Reconstruction using the narrow box image as reference. Note the false reconstructions. This image was produced after a sequence of alternating subject reference pairs had been presented to partially train the system in differential recognition.



B. Reconstruction with wide box image after partial training. Note false reconstructions of N near the W. Note also the false reconstructions due to the other corners of the box.



C. Reconstruction with training almost completed. The last iteration sequence had used a W-wide box combination. This explains the residual cross talk.



D. Reconstruction with wide box reference and the system in the same state of training as view C.

The system has learned to distinguish between reference patterns that have a 0.689 correlation.

FIGURE 15. RESULTS OF DIFFERENTIAL TRAINING EXPERIMENT

93-8-16

NOTE: The software included in this document is configured for a Data General Eclipse Computer supported with a 100 M Byte disk memory.

SECTION 4

MACHINE HOLOGRAPHY SOFTWARE

4.1 SUMMARY

The machine holography exercise explored the data storage properties and some image recognition properties of holograms through use of a computer simulation. The software developed for the hologram simulation consists of five main classes of programs:

(1) Image creation routines

(2) Hologram routines

(3) I/O routines

(4) Utility routines

(5) Borrowed and miscellaneous routines

All machine holography software resides on disk under directory HOLO.DR. The directory is saved on magnetic tape. File 0 of the tape contains the holography programs, and File 1 contains the holography image files.

The image creation routines create various simple point images and random noise scenes in the integer disk file format used by the machine holography software. The image creation routines have file names of the following form: CR—.FR. All image creation software is a variation to the CRSCN.FR program which creates a point image at specified coordinates in an integer array stored on disk. CRNOI.— program creates random noise integer arrays in the standard disk format used by machine holography software. There appeared to be some correlation between different random scenes, so the random scenes were input to the program SCRAM.FR to "scramble" (randomly shift) the rows and columns of the data file. The following are scene creation routines:

CRSCN.—	Creates single point image file; ex. file P3333. is a point at coordinates X = 33 Y = 33
CRNO1.—	Creates a random noise scene; ex. file NOISE1 is a random noise file
SCRAM.—	Used to "randomize" a random scene. This program modifies a file that had been previously created.
CR4SCN.—	Creates a four point image; ex. file F3232 is a file of four points with LH corner at X = 32, Y = 32

CR16SCN.— Creates a sixteen point scene; ex. S3232 is a file of sixteen points with LH corner at X=32, Y=32

CRLNO1.— Creates a random noise scene of limited size (L64x64)

CRN.— Creates an image "N"; ex. file IMAGEN was created by CRN.SV

CRW.— Creates an image of "W"; ex. file IMAGEW was created by CRW.SV

CR3.—
CR4.—
CR5.—
CR6.— Creates an image file of a "+" and an integer. These were used to create the Tank I.D. files: TID1, TID2, TID3, TID4, TID5, and TID6.

CRSQ.— Creates a square image

All image files can be examined by executing the program IMAGER.SV. IMAGER reads the image from disk and outputs a grayscale plot to the line printer. The subroutine SDEF (contained in disk file IREAD.—) is used by machine holography software to read the image files from disk into memory.

The actual holography routines consist of four programs: HDRV.—, HSDRV.— MHOLO.— FBORVT.—, and CRSUM.—.

HDRV.— Creates a single hologram and performs an image reconstruction

HSDRV.— Creates a multiplexed hologram

MHOLO.— Reads a hologram, performs an image reconstruction, and outputs a grayscale plot. MHOLO. is used in conjunction with HSDRV.

FBDRVT.— Creates a hologram feedback loop

CRSUM.— Is a utility program used to zero a complex array file on disk. CRSUM. is executed to create a zero valued hologram file prior to running HSDRV.—

I/O routines are RBLC.—, RBLKC.—, WBLC.—, and WBLKC.—. These routines are used to read and write 64 by 64 complex arrays to and from disk.

Several routines were borrowed from different directories. These are FFT2D.RB, GSPLT.—, PICPLT.RB, and PLPT.RB.

FFT2D.RB	Performs a two-dimensional Fourier transform on a complex array
GSPLT.—	Performs a grayscale plot of a complex data array. The program was modified to calculate S/N of the array.
GSPLTX.—	Is a modified version of GSPLT.— which has a parameter in the call statement specifying how many points are used in the S/N calculation
PLPT.RB and PICPLT.RB	Are used by GSPLT and GSPLTX

The software operates on 64 by 64 complex arrays. Because of limited memory, as the programs execute they swap data back and forth from memory to disk. The driver programs create files and read existing files. The operator specifies the required file names when prompted by the programs. The file names are stored in seven element integer arrays, so file names are limited to seven characters. Any I/O error causes the programs to stop.

The utility subroutines and the disk I/O subroutines (except SUBROUTINE SDEF) use the format shown in Figure 16 to read and write data. The disk file uses sixty-five 256-word blocks.

NOTE:

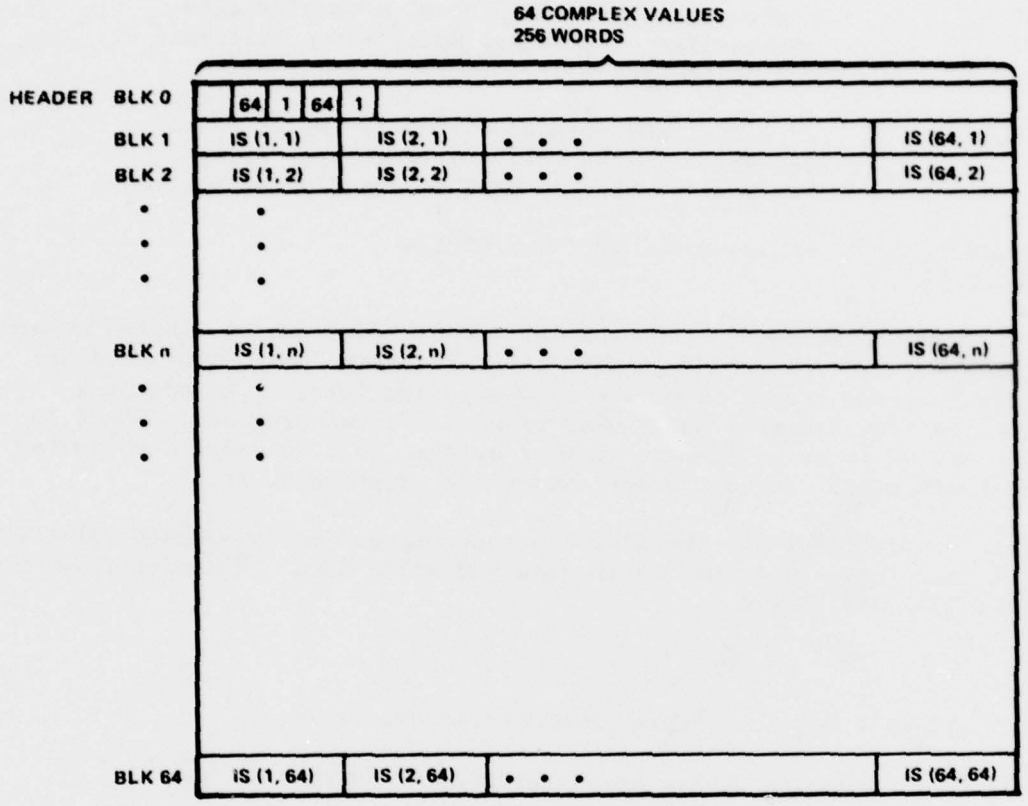
(1) All file names used are seven characters or less.

(2) File names are stored by the program in integer arrays.

(3) Data stored on disk files are in standard format, i.e., either integer format or complex format

(4) In the following write-up, upper case letters correspond to program console output. Underlines lowercase letters correspond to operator console input.

(5) ↵ means carriage return.



- DATA IS STORED IN MEMORY IN THE COMPLEX ARRAY IS (64, 64)

COLUMN INDEX — ROW INDEX

- DATA IS STORED IN 65 BLOCKS ON THE DISK FILE. EACH BLOCK CONTAINS ONE ROW OF DATA (64 COMPLEX VALUES · 256 WORDS)

93-8-1

FIGURE 16. DATA FORMAT OF COMPLEX DISK FILE DATA

4.2 IMAGE CREATION ROUTINES

All images and scenes used by the machine holography software are stored on integer disk files according to the format in Figure 17. The disk file stores a two-dimensional array. Each row of the array is stored in one block of the file with consecutive rows stored in consecutive blocks. Each block is 256 words, with each row of the image having 256 integer values. All machine holography images are 64 by 64, so only a quarter of each block is used. This is shown by the shaded area in Figure 17. The image file can have an arbitrary number of rows (blocks) of data. Machine holography images have 64 rows of data. All machine holography images have array coordinates 1,1 at the upper left hand corner of image array file. All integer files are read using IMAGER.SV (for plots) or SUB SDEF (contained in file IREAD.—).

4.2.1 CRSCN.FR

CRSCN.FR is a FORTRAN program that generates an integer image file. All machine holography programs of the form CR—.FR (except CRSUM) are modifications of CRSCN.FR.

CRSCN.FR creates an integer image file consisting of a single point of a specified value, at specified coordinates, with the remaining points of the image set to zero. All CR—.FR programs (except CRSUM) ask for the image array size and position within the data file (refer to Figure 17). All machine holography images are currently 64 by 64 in size and are positioned with the upper left-hand corner positioned at coordinates 1,1.

Modifications to CRSCN., in order to create images other than a single point, are made by modifying the FORTRAN code in the DO 5Ø loop. This can be best understood by examining the listings of the different CR—.FR programs. The following is the dialogue between the program and operator.

```
RUN  n
COLUMN  integer
ROW  integer
```

(NOTE: Column and row are the coordinates of the image point within the image array.)

```
VALUE  real number
```

(NOTE: Value is the "intensity" of the point. VALUE = 100. for most machine holography images.)

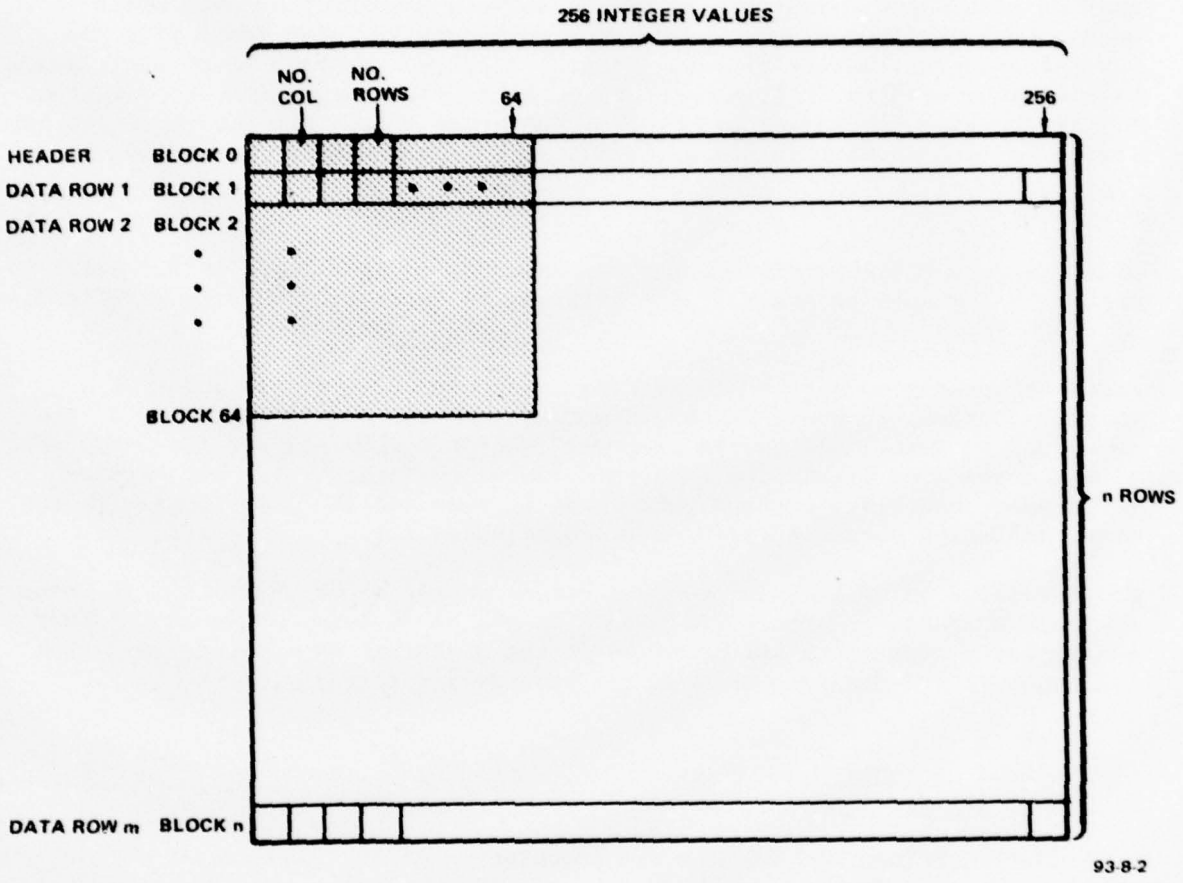


FIGURE 17. INTEGER IMAGE FORMAT

```
C   CRSCN.FR        7/05/78
C
C       CRSCN GENERATES AN 256 X JRMX ARRAY CONTAINING
C       AN NX X NY IMAGE WITH UPPER LH CORNER COORDS (NXC,NYC).
C       THE PORTIONS OF THE ARRAY NOT CONTAINING THE IMAGE ARE
C       FILLED WITH ZEROS AND THE ARRAY IS STORED ON DISK AS KNAME
C
        DIMENSION IDD(256),KNAME(7)

C
C       KNAME IN FILE #1 IS USED TO STORE THE BACKGROUND IMAGE

        DO 80 I80=1,10
        TYPE "RUN ",I80
        ACCEPT "COLUMN ",NCOL,"ROW ",NROW,"VALUE ",VALUE
        WRITE(10,900)
900     FORMAT(" NAME OF FILE TO BE CREATED = ",Z)
        READ(11,905) KNAME(1)
905     FORMAT(S7)
        CALL DFILW(KNAME(1),IER)
        CALL CFILW(KNAME(1),2,IER)
        IF(IER.NE.1) STOP -- CFILW ERROR (BGD) --
        CALL OPEN(1,KNAME(1),2,IER,512)
        IF(IER.NE.1) STOP --OPEN ERROR (BGD) --
C
C       INPUT IMAGE PARAMETERS
C
        ACCEPT"IMAGE SIZE (NX,NY) = ",NX,NY
        TYPE"COORDS OF UPPER LH CORNER OF IMAGE"
        TYPE"E.G.  (1,1)= UPPER LH CORNER OF ENTIRE ARRAY"
        ACCEPT"(NXC,NYC) = ",NXC,NYC
        JCMX=NX+NXC-1
        JRMX=NY+NYC-1
        IF(JCMX .GT. 256) STOP--IMAGE OFF RIGHT EDGE OF ARRAY--

        PI=3.1415926
        ARG=PI/8.
        DO 10 I=1,256
10      IDD(I)=0
        IDD(2)=256
        IDD(4)=JRMX
        CALL WRBLK(1,0,IDD,1,IER)
        IF(IER .NE. 1)STOP--WRBLK ERR--
        IDD(2)=0
        IDD(4)=0
        NS1=NYC-1
                                                IF(NS1 .LE. 0) GO TO 30
        DO 20 IR=1,NS1
20      CALL WRBLK(1,IR,IDD,1,IER)
        IF(IER .NE. 1) STOP--WRBLK ERR--
30      CONTINUE
        RMEAN=0
        SIGMA=256
        NRAND=2351
        DO 50 IR=NYC,JRMX
        IIR=IR+NYC-1
        DO 5 IC=1,NX
        IIC=IC+NXC-1
        IDD(IIC)=0
5       CONTINUE
        IF(IR .EQ. NROW)IDD(NCOL)=VALUE
        CALL WRBLK(1,IR,IDD,1,IER)
        IF(IER.NE.1) STOP --WRBLK ERROR (BGD)--
50      CONTINUE

        CALL CLOSE(1,IER)
   80   CONTINUE
        STOP
        END
```

```
C   CRNOI.FR          7/05/78
C
C       CRSCN GENERATES AN 256 X JRMX ARRAY CONTAINING
C       AN NX X NY IMAGE WITH UPPER LH CORNER COORDS (NXC,NYC).
C       THE PORTIONS OF THE ARRAY NOT CONTAINING THE IMAGE ARE
C       FILLED WITH ZEROS AND THE ARRAY IS STORED ON DISK AS KNAME
C
        DIMENSION IDD(256),KNAME(7)

C
C       KNAME IN FILE #1 IS USED TO STORE THE BACKGROUND IMAGE

        RMEAN=0.
        SIGMA=256.
        NRAND=2351
        ACCEPT "RANDOM SEED",NRAND
        DO 80 I80=1,10
        TYPE "RUN ",I80
        WRITE(10,900)
900     FORMAT(" NAME OF FILE TO BE CREATED = ",Z)
        READ(11,905) KNAME(1)
905     FORMAT(S7)
        CALL DFILW(KNAME(1),IER)
        CALL CFILW(KNAME(1),2,IER)
        IF(IER.NE.1) STOP -- CFILW ERROR (BGD) --
        CALL OPEN(1,KNAME(1),2,IER,512)
        IF(IER.NE.1) STOP --OPEN ERROR (BGD) --
C
C       INPUT IMAGE PARAMETERS
C
        ACCEPT"IMAGE SIZE (NX,NY) = ",NX,NY
        TYPE'COORDS OF UPPER LH CORNER OF IMAGE"
        TYPE'E.G.  (1,1)= UPPER LH CORNER OF ENTIRE ARRAY"
        ACCEPT"(NXC,NYC) = ",NXC,NYC
        JCMX=NX+NXC-1
        JRMX=NY+NYC-1
        IF(JCMX.GT.256) STOP--IMAGE OFF RIGHT EDGE OF ARRAY--

        PI=3.1415926
        ARG=PI/8
        DO 10 I=1,256
10      IDD(I)=0
        IDD(2)=256
        IDD(4)=JRMX
        CALL WRBLK(1,0,IDD,1,IER)
        IF(IER.NE.1)STOP--WRBLK ERR--
        IDD(2)=0
        IDD(4)=0
        NS1=NYC-1
        IF(NS1.LE.0) GO TO 30
        DO 20 IR=1,NS1
20      CALL WRBLK(1,IR,IDD,1,IER)
        IF(IER.NE.1) STOP--WRBLK ERR--
30      CONTINUE
        DO 50 IR=NYC,JRMX
        IIR=IR+NYC-1
        DO 5 IC=1,NX
        IIC=IC+NXC-1
        IDD(IIC)=RGAUS(RMEAN,SIGMA,NRAND)
5       CONTINUE
        CALL WRBLK(1,IR,IDD,1,IER)
        IF(IER.NE.1) STOP --WRBLK ERROR (BGD)--
50      CONTINUE

        CALL CLOSE(1,IER)
   80   CONTINUE
        STOP
        END
```

```
C
C       THIS ROUTINE RANDOMIZES A RANDOM ARRAY
C       BY PERFORMING RANDOM CIRCULAR SHIFTS
C       OF THE ROWS AND COLUMNS
C
C       SUBROUTINES USED:          RAND.RB
C                                  IREAD.RB
C
        DIMENSION IDD(256),KNAME(7)
        COMPLEX IS(64,64),TEMP(64)
        ACCEPT "INPUT SEED FOR RANDOM NUMBER",NRND
        IMN=0
        IAT=1
        M=6
        NR=64
        NC=64
        CALL SDEF(IS,M,IMN,IAT,KNAME)
C
C       SHIFT AND INTERCHANGE COLUMNS
C
        DO 50 I50=1,NR
        CALL RAND(NRND)
        INDEX=FLOAT(NRND)*64./65536.
        INDEX=INDEX+32
C
        TYPE INDEX,INDEX,INDEX
C
        DO 40 I40=1,NC
        MOVE=1+MOD(I40+INDEX,64)
        TEMP(I40)=IS(MOVE,I50)
   40   CONTINUE
        DO 41 I41=1,NC
        IS(I41,I50)=TEMP(I41)
   41   CONTINUE
   50   CONTINUE
C
C       SHIFT AND INTERCHANGE ROWS
C
        DO 60 I60=1,NC
        CALL RAND(NRND)
        INDEX=FLOAT(NRND)*64./65536.
        INDEX=INDEX+32
C
        TYPE INDEX,INDEX,INDEX
C
        DO 70 I70=1,NR
        MOVE=1+MOD(I70+INDEX,64)
        TEMP(I70)=IS(I60,MOVE)
   70   CONTINUE
        DO 71 I71=1,NR
        IS(I60,I71)=TEMP(I71)
   71   CONTINUE
   60   CONTINUE
C
C
C       WRITE FILE TO DISK
C
        WRITE(10,900)
  900   FORMAT("  NAME OF FILE TO BE CREATED = ",Z)
        READ(11,905)KNAME(1)
  905   FORMAT(S7)
        CALL DFILW(KNAME(1),IER)
        CALL CFILW(KNAME(1),2,IER)
        IF(IER .NE. 1)STOP --CREATE FILE ERROR--
        CALL OPEN(1,KNAME(1),2,IER,512)
        IF(IER.NE.1) STOP --OPEN ERROR--
        NX=64
        NY=64
        NXC=1
        NYC=1
        JCMX=NX+NXC-1
        JRMX=NY+NYC-1
C
        DO 10 I10=1,256
        IDD(I10)=0
   10   CONTINUE
        IDD(2)=256
        IDD(4)=JRMX
C
        CALL WRBLK(1,0,IDD,1,IER)
        IF(IER .NE. 1)STOP --WRBLK ERROR--
        IDD(2)=0
        IDD(4)=0
C
        DO 100 IR=NYC,JRMX
        IIR=IR+NYC-1
        DO 5 IC=1,NX
        IIC=IC+NXC-1
        IDD(IIC)=REAL(IS(IC,IR))
    5   CONTINUE
        CALL WRBLK(1,IR,IDD,1,IER)
        IF(IER .NE. 1)STOP --WRBLK ERROR--
  100   CONTINUE
C
        CALL CLOSE(1,IER)
        STOP
        END
```

```
C   CR4SCN.FR          11/09/78
C
C       CRSCN GENERATES AN 256 X JRMX ARRAY CONTAINING
C       AN NX X NY IMAGE WITH UPPER LH CORNER COORDS (NXC,NYC).
C       THE PORTIONS OF THE ARRAY NOT CONTAINING THE IMAGE ARE
C       FILLED WITH ZEROS AND THE ARRAY IS STORED ON DISK AS KNAME
C
        DIMENSION IDD(256),KNAME(7)

C
C       KNAME IN FILE #1 IS USED TO STORE THE BACKGROUND IMAGE

        DO 80 I80=1,16
        TYPE "RUN ",I80
        ACCEPT "COLUMN ",NCOL,"ROW ",NROW,"VALUE ",VALUE
        WRITE(10,900)
900     FORMAT(" NAME OF FILE TO BE CREATED = ",Z)
        READ(11,905) KNAME(1)
905     FORMAT(S7)
        CALL DFILW(KNAME(1),IER)
        CALL CFILW(KNAME(1),2,IER)
        IF(IER.NE.1) STOP -- CFILW ERROR (BGD) --
        CALL OPEN(1,KNAME(1),2,IER,512)
        IF(IER.NE.1) STOP --OPEN ERROR (BGD) --
C
C       INPUT IMAGE PARAMETERS
C
        ACCEPT"IMAGE SIZE (NX,NY) = ",NX,NY
        TYPE"COORDS OF UPPER LH CORNER OF IMAGE"
        TYPE"E.G.  (1,1)= UPPER LH CORNER OF ENTIRE ARRAY"
        ACCEPT"(NXC,NYC) = ",NXC,NYC
        JCMX=NX+NXC-1
        JRMX=NY+NYC-1
        IF(JCMX.GT.256) STOP--IMAGE OFF RIGHT EDGE OF ARRAY--

        PI=3.1415926
        ARG=PI/8
        DO 10 I=1,256
10      IDD(I)=0
        IDD(2)=256
        IDD(4)=JRMX
        CALL WRBLK(1,0,IDD,1,IER)
        IF(IER.NE.1)STOP--WRBLK ERR--
        IDD(2)=0
        IDD(4)=0
        NS1=NYC-1
        IF(NS1.LE.0) GO TO 30
        DO 20 IR=1,NS1
20      CALL WRBLK(1,IR,IDD,1,IER)
        IF(IER.NE.1) STOP--WRBLK ERR--
30      CONTINUE
        RMEAN=0
        SIGMA=256
        NRAND=2351
        DO 50 IR=NYC,JRMX
        IIR=IR+NYC-1
        DO 5 IC=1,NX
        IIC=IC+NXC-1
        IDD(IIC)=0
5       CONTINUE
        IF(IR.EQ.NROW)IDD(NCOL)=VALUE
        IF(IR.EQ.NROW)IDD(NCOL+3)=VALUE
        IF(IR.EQ.NROW+3)IDD(NCOL)=VALUE
        IF(IR.EQ.NROW+3)IDD(NCOL+3)=VALUE
        CALL WRBLK(1,IR,IDD,1,IER)
        IF(IER.NE.1) STOP --WRBLK ERROR (BGD)--
50      CONTINUE

        CALL CLOSE(1,IER)
   80   CONTINUE
        STOP
        END
```

```
C   CR16SCN FR          11/09/78
C
C       CRSCN GENERATES AN 256 X JRMX ARRAY CONTAINING
C       AN NX X NY IMAGE WITH UPPER LH CORNER COORDS (NXC,NYC).
C       THE PORTIONS OF THE ARRAY NOT CONTAINING THE IMAGE ARE
C       FILLED WITH ZEROS AND THE ARRAY IS STORED ON DISK AS KNAME
C
        DIMENSION IDD(256),KNAME(7)

C
C       KNAME IN FILE #1 IS USED TO STORE THE BACKGROUND IMAGE

        DO 80 I80=1,16
        TYPE "RUN ",I80
        ACCEPT "COLUMN ",NCOL,"ROW ",NROW,"VALUE ",VALUE
        WRITE(10,900)
900     FORMAT(" NAME OF FILE TO BE CREATED = ",Z)
        READ(11,905) KNAME(1)
905     FORMAT(S7)
        CALL DFILW(KNAME(1),IER)
        CALL CFILW(KNAME(1),2,IER)
        IF(IER.NE.1) STOP -- CFILW ERROR (BGD) --
        CALL OPEN(1,KNAME(1),2,IER,512)
        IF(IER.NE.1) STOP --OPEN ERROR (BGD) --
C
C       INPUT IMAGE PARAMETERS
C
        ACCEPT"IMAGE SIZE (NX,NY) = ",NX,NY
        TYPE"COORDS OF UPPER LH CORNER OF IMAGE"
        TYPE"E.G. (1,1)= UPPER LH CORNER OF ENTIRE ARRAY"
        ACCEPT"(NXC,NYC) = ",NXC,NYC
        JCMX=NX+NXC-1
        JRMX=NY+NYC-1
        IF(JCMX.GT.256) STOP--IMAGE OFF RIGHT EDGE OF ARRAY--

        PI=3.1415926
        ARG=PI/8.
        DO 10 I=1,256
10      IDD(I)=0
        IDD(2)=256
        IDD(4)=JRMX
        CALL WRBLK(1,0,IDD,1,IER)
        IF(IER.NE.1)STOP--WRBLK ERR--
        IDD(2)=0
        IDD(4)=0
        NS1=NYC-1
        IF(NS1.LE.0) GO TO 30
        DO 20 IR=1,NS1
20      CALL WRBLK(1,IR,IDD,1,IER)
        IF(IER.NE.1) STOP--WRBLK ERR--
30      CONTINUE
        RMEAN=0
        SIGMA=256
        NRAND=2351
        DO 50 IR=NYC,JRMX
        IIR=IR+NYC-1
        DO 5 IC=1,NX
        IIC=IC+NXC-1
        IDD(IIC)=0
5       CONTINUE
        IF(IR.EQ.NROW)IDD(NCOL)=VALUE
        IF(IR.EQ.NROW)IDD(NCOL+4)=VALUE
        IF(IR.EQ.NROW)IDD(NCOL+8)=VALUE
        IF(IR.EQ.NROW)IDD(NCOL+12)=VALUE
        IF(IR.EQ.NROW+4)IDD(NCOL)=VALUE
        IF(IR.EQ.NROW+4)IDD(NCOL+4)=VALUE
        IF(IR.EQ.NROW+4)IDD(NCOL+8)=VALUE
        IF(IR.EQ.NROW+4)IDD(NCOL+12)=VALUE
        IF(IR.EQ.NROW+8)IDD(NCOL)=VALUE
        IF(IR.EQ.NROW+8)IDD(NCOL+4)=VALUE
        IF(IR.EQ.NROW+8)IDD(NCOL+8)=VALUE
        IF(IR.EQ.NROW+8)IDD(NCOL+12)=VALUE
        IF(IR.EQ.NROW+12)IDD(NCOL)=VALUE
        IF(IR.EQ.NROW+12)IDD(NCOL+4)=VALUE
        IF(IR.EQ.NROW+12)IDD(NCOL+8)=VALUE
        IF(IR.EQ.NROW+12)IDD(NCOL+12)=VALUE
        CALL WRBLK(1,IR,IDD,1,IER)
        IF(IER.NE.1) STOP --WRBLK ERROR (BGD)--
50      CONTINUE

        CALL CLOSE(1,IER)
   80   CONTINUE
        STOP
        END
```

```
C    CRLNOI.FR          11/10/78
CC
CC        CRSCN GENERATES AN 256 X JRMX ARRAY CONTAINING
CC        AN NX X NY IMAGE WITH UPPER LH CORNER COORDS (NXC,NYC).
CC        THE PORTIONS OF THE ARRAY NOT CONTAINING THE IMAGE ARE
CC        FILLED WITH ZEROS AND THE ARRAY IS STORED ON DISK AS KNAME
C
          DIMENSION IDD(256),KNAME(7)

C
C         KNAME IN FILE #1 IS USED TO STORE THE BACKGROUND IMAGE

          RMEAN=0.
          SIGMA=256.
          NRAND=2351
          ACCEPT "RANDOM SEED",NRAND
          DO 80 I80=1,10
          TYPE "RUN ",I80
          ACCEPT "REFERENCE SIZE = ",NSIZE
          ACCEPT "REFERENCE POSITION = ",NPOS
          WRITE(10,900)
900       FORMAT(" NAME OF FILE TO BE CREATED = ",Z)
          READ(11,905) KNAME(1)
905       FORMAT(S7)
          CALL DFILW(KNAME(1),IER)
          CALL CFILW(KNAME(1),2,IER)
          IF(IER.NE.1) STOP -- CFILW ERROR (BGD) --
          CALL OPEN(1,KNAME(1),2,IER,512)
          IF(IER.NE.1) STOP --OPEN ERROR (BGD) --
C
CC        INPUT IMAGE PARAMETERS
C
          ACCEPT"IMAGE SIZE (NX,NY) = ",NX,NY
          TYPE"COORDS OF UPPER LH CORNER OF IMAGE"
          TYPE"E.G.  (1,1)= UPPER LH CORNER OF ENTIRE ARRAY"
          ACCEPT"(NXC,NYC) = ",NXC,NYC
          JCMX=NX+NXC-1
          JRMX=NY+NYC-1
          IF(JCMX .GT. 256) STOP--IMAGE OFF RIGHT EDGE OF ARRAY--

          PI=3.1415926
          ARG=PI/8.
          DO 10 I=1,256
10        IDD(I)=0
          IDD(2)=256
          IDD(4)=JRMX
          CALL WRBLK(1,0,IDD,1,IER)
          IF(IER .NE. 1)STOP--WRBLK ERR--
          IDD(2)=0
          IDD(4)=0
          NS1=NYC-1
          IF(NS1 .LE. 0) GO TO 30
          DO 20 IR=1,NS1
20        CALL WRBLK(1,IR,IDD,1,IER)
          IF(IER .NE. 1) STOP--WRBLK ERR--
30        CONTINUE
          DO 50 IR=NYC,JRMX
          IIR=IR+NYC-1
          IF(IIR .LT. 28 .OR. IIR .GE. 34)GO TO 555
          DO 5 IC=1,NX
          IIC=IC+NXC-1
          IF(IIC .EQ. 29 .OR. IIC .EQ. 33)IDD(IIC)=100
          IF(IIR .EQ. 30 .AND. IIC .EQ. 30)IDD(IIC)=100
          IF(IIR .EQ. 31 .AND. IIC .EQ. 31)IDD(IIC)=100
          IF(IIR .EQ. 32 .AND. IIC .EQ. 32)IDD(IIC)=100
  554     CONTINUE
5         CONTINUE
  555     CONTINUE
          CALL WRBLK(1,IR,IDD,1,IER)
          IF(IER.NE.1) STOP --WRBLK ERROR (BGD)--
          DO 6 I6=1,NX
          IDD(I6)=0
    6     CONTINUE
  556     CONTINUE
50        CONTINUE

          CALL CLOSE(1,IER)
   80     CONTINUE
          STOP
          END
```

```
C   CRW FR          12/20/78
C
C       CRSCN GENERATES AN 256 X JRMX ARRAY CONTAINING
C       AN NX X NY IMAGE WITH UPPER LH CORNER COORDS (NXC,NYC)
C       THE PORTIONS OF THE ARRAY NOT CONTAINING THE IMAGE ARE
C       FILLED WITH ZEROS AND THE ARRAY IS STORED ON DISK AS KNAME
C
        DIMENSION IDD(256),KNAME(7)

C
C       KNAME IN FILE #1 IS USED TO STORE THE BACKGROUND IMAGE

        RMEAN=0
        SIGMA=256
        NRAND=2351
        ACCEPT "RANDOM SEED",NRAND
        DO 80 I80=1,10
        TYPE "RUN ",I80
        ACCEPT "REFERENCE SIZE = ",NSIZE
        ACCEPT "REFERENCE POSITION = ",NPOS
        WRITE(10,900)
900     FORMAT(" NAME OF FILE TO BE CREATED = ",Z)
        READ(11,905) KNAME(1)
905     FORMAT(S7)
        CALL DFILW(KNAME(1),IER)
        CALL CFILW(KNAME(1),2,IER)
        IF(IER NE 1) STOP -- CFILW ERROR (BGD) --
        CALL OPEN(1,KNAME(1),2,IER,512)
        IF(IER NE 1) STOP --OPEN ERROR (BGD) --
C
C       INPUT IMAGE PARAMETERS
C
        ACCEPT"IMAGE SIZE (NX,NY) = ",NX,NY
        TYPE"COORDS OF UPPER LH CORNER OF IMAGE"
        TYPE"E G  (1,1)= UPPER LH CORNER OF ENTIRE ARRAY"
        ACCEPT"(NXC,NYC) = ",NXC,NYC
        JRMX=NX+NXC-1
        JRMX=NY+NYC-1
        IF(JRMX  GT  256) STOP--IMAGE OFF RIGHT EDGE OF ARRAY--

        PI=3.1415926
        ARG=PI/8
        DO 10 I=1,256
10      IDD(I)=0
        IDD(2)=256
        IDD(4)=JRMX
        CALL WRBLK(1,0,IDD,1,IER)
        IF(IER  NE  1)STOP--WRBLK ERR--
        IDD(2)=0
        IDD(4)=0
        NS1=NYC-1
        IF(NS1  LE  0) GO TO 30
        DO 20 IR=1,NS1
20      CALL WRBLK(1,IR,IDD,1,IER)
        IF(IER  NE  1) STOP--WRBLK ERR--
30      CONTINUE
        DO 50 IR=NYC,JRMX
        IIR=IR+NYC-1
        IF(IIR  LT  29  OR  IIR  GT  35)GO TO 555
        DO 5 IC=1,NX
        IIC=IC+NXC-1
        IDD(32)=100
        IF(IIR  EQ  32  AND  IIC  EQ  30)IDD(IIC)=100
        IF(IIR  EQ  32  AND  IIC  EQ  31)IDD(IIC)=100
        IF(IIR  EQ  32  AND  IIC  EQ  33)IDD(IIC)=100
        IF(IIR  EQ  32  AND  IIC  EQ  34)IDD(IIC)=100
        IF(IIR  EQ  32  AND  IIC  EQ  29)IDD(IIC)=100
        IF(IIR  EQ  32  AND  IIC  EQ  35)IDD(IIC)=100
C
        IF(IIR  EQ  29  AND  IIC  EQ  49)IDD(IIC)=100
        IF(IIR  EQ  29  AND  IIC  EQ  50)IDD(IIC)=100
        IF(IIR  EQ  29  AND  IIC  EQ  51)IDD(IIC)=100
        IF(IIR  EQ  30  AND  IIC  EQ  51)IDD(IIC)=100
        IF(IIR  EQ  31  AND  IIC  EQ  51)IDD(IIC)=100
        IF(IIR  EQ  32  AND  IIC  EQ  49)IDD(IIC)=100
        IF(IIR  EQ  32  AND  IIC  EQ  50)IDD(IIC)=100
        IF(IIR  EQ  32  AND  IIC  EQ  51)IDD(IIC)=100
        IF(IIR  EQ  33  AND  IIC  EQ  51)IDD(IIC)=100
        IF(IIR  EQ  34  AND  IIC  EQ  51)IDD(IIC)=100
        IF(IIR  EQ  35  AND  IIC  EQ  49)IDD(IIC)=100
        IF(IIR  EQ  35  AND  IIC  EQ  50)IDD(IIC)=100
        IF(IIR  EQ  35  AND  IIC  EQ  51)IDD(IIC)=100
C
  554   CONTINUE
5       CONTINUE
  555   CONTINUE
        CALL WRBLK(1,IR,IDD,1,IER)
        IF(IER NE 1) STOP --WRBLK ERROR (BGD)--
        DO 6 I6=1,NX
        IDD(I6)=0
    6   CONTINUE
  556   CONTINUE
50      CONTINUE

        CALL CLOSE(1,IER)
   80   CONTINUE
        STOP
        END
```

```
C    CR3              12/20/78
C
C         CRSCN GENERATES AN 256 X JRMX ARRAY CONTAINING
C         AN NX X NY IMAGE WITH UPPER LH CORNER COORDS (NXC,NYC).
C         THE PORTIONS OF THE ARRAY NOT CONTAINING THE IMAGE ARE
C         FILLED WITH ZEROS AND THE ARRAY IS STORED ON DISK AS KNAME
C
          DIMENSION IDD(256),KNAME(7)

C
C         KNAME IN FILE #1 IS USED TO STORE THE BACKGROUND IMAGE

          RMEAN=0.
          SIGMA=256.
          NRAND=2351
          ACCEPT "RANDOM SEED",NRAND
          DO 80 I80=1,10
          TYPE "RUN ",I80
          ACCEPT "REFERENCE SIZE = ",NSIZE
          ACCEPT "REFERENCE POSITION = ",NPOS
          WRITE(10,900)
900       FORMAT(" NAME OF FILE TO BE CREATED = ",Z)
          READ(11,905) KNAME(1)
905       FORMAT(S7)
          CALL DFILW(KNAME(1),IER)
          CALL CFILW(KNAME(1),2,IER)
          IF(IER.NE.1) STOP -- CFILW ERROR (BGD) --
          CALL OPEN(1,KNAME(1),2,IER,512)
          IF(IER.NE.1) STOP --OPEN ERROR (BGD) --
C
C         INPUT IMAGE PARAMETERS
C
          ACCEPT"IMAGE SIZE (NX,NY) = ",NX,NY
          TYPE"COORDS OF UPPER LH CORNER OF IMAGE"
          TYPE"E.G.  (1,1)= UPPER LH CORNER OF ENTIRE ARRAY"
          ACCEPT"(NXC,NYC) = ",NXC,NYC
          JCMX=NX+NXC-1
          JRMX=NY+NYC-1
          IF(JCMX .GT. 256) STOP--IMAGE OFF RIGHT EDGE OF ARRAY--

          PI=3.1415926
          ARG=PI/8.
          DO 10 I=1,256
10        IDD(I)=0
          IDD(2)=256
          IDD(4)=JRMX
          CALL WRBLK(1,0,IDD,1,IER)
          IF(IER .NE. 1)STOP--WRBLK ERR--
          IDD(2)=0
          IDD(4)=0
          NS1=NYC-1
          IF(NS1 .LE. 0) GO TO 30
          DO 20 IR=1,NS1
20        CALL WRBLK(1,IR,IDD,1,IER)
          IF(IER .NE. 1) STOP--WRBLK ERR--
30        CONTINUE
          DO 50 IR=NYC,JRMX
          IIR=IR+NYC-1
          IF(IIR .LT. 29 .OR. IIR .GT. 35)GO TO 555
          DO 5 IC=1,NX
          IIC=IC+NXC-1
          IDD(32)=100
          IF(IIR .EQ. 32 .AND. IIC .EQ. 30)IDD(IIC)=100
          IF(IIR .EQ. 32 .AND. IIC .EQ. 31)IDD(IIC)=100
          IF(IIR .EQ. 32 .AND. IIC .EQ. 33)IDD(IIC)=100
          IF(IIR .EQ. 32 .AND. IIC .EQ. 34)IDD(IIC)=100
          IF(IIR .EQ. 32 .AND. IIC .EQ. 29)IDD(IIC)=100
          IF(IIR .EQ. 32 .AND. IIC .EQ. 35)IDD(IIC)=100
C
          IF(IIR .EQ. 29 .AND. IIC .EQ. 49)IDD(IIC)=100
          IF(IIR .EQ. 29 .AND. IIC .EQ. 50)IDD(IIC)=100
          IF(IIR .EQ. 29 .AND. IIC .EQ. 51)IDD(IIC)=100
          IF(IIR .EQ. 30 .AND. IIC .EQ. 51)IDD(IIC)=100
          IF(IIR .EQ. 31 .AND. IIC .EQ. 51)IDD(IIC)=100
          IF(IIR .EQ. 32 .AND. IIC .EQ. 49)IDD(IIC)=100
          IF(IIR .EQ. 32 .AND. IIC .EQ. 50)IDD(IIC)=100
          IF(IIR .EQ. 32 .AND. IIC .EQ. 51)IDD(IIC)=100
          IF(IIR .EQ. 33 .AND. IIC .EQ. 51)IDD(IIC)=100
          IF(IIR .EQ. 34 .AND. IIC .EQ. 51)IDD(IIC)=100
          IF(IIR .EQ. 35 .AND. IIC .EQ. 49)IDD(IIC)=100
          IF(IIR .EQ. 35 .AND. IIC .EQ. 50)IDD(IIC)=100
          IF(IIR .EQ. 35 .AND. IIC .EQ. 51)IDD(IIC)=100
C
  554     CONTINUE
5         CONTINUE
  555     CONTINUE
          CALL WRBLK(1,IR,IDD,1,IER)
          IF(IER.NE.1) STOP --WRBLK ERROR (BGD)--
          DO 6 I6=1,NX
          IDD(I6)=0
    6     CONTINUE
  556     CONTINUE
50        CONTINUE

          CALL CLOSE(1,IER)
   80     CONTINUE
          STOP
          END
```

```
C   CR4                12/20/78
C
C       CRSCN GENERATES AN 256 X JRMX ARRAY CONTAINING
C       AN NX X NY IMAGE WITH UPPER LH CORNER COORDS (NXC,NYC)
C       THE PORTIONS OF THE ARRAY NOT CONTAINING THE IMAGE ARE
C       FILLED WITH ZEROS AND THE ARRAY IS STORED ON DISK AS KNAME
C
        DIMENSION IDD(256),KNAME(7)

C
C       KNAME IN FILE #1 IS USED TO STORE THE BACKGROUND IMAGE

        RMEAN=0.
        SIGMA=256.
        NRAND=2351
        ACCEPT "RANDOM SEED",NRAND
        DO 80 I80=1,10
        TYPE "RUN ",I80
        ACCEPT "REFERENCE SIZE = ",NSIZE
        ACCEPT "REFERENCE POSITION = ",NPOS
        WRITE(10,900)
900     FORMAT(" NAME OF FILE TO BE CREATED = ",Z)
        READ(11,905) KNAME(1)
905     FORMAT(S7)
        CALL DFILW(KNAME(1),IER)
        CALL CFILW(KNAME(1),2,IER)
        IF(IER.NE.1) STOP -- CFILW ERROR (BGD) --
        CALL OPEN(1,KNAME(1),2,IER,512)
        IF(IER.NE.1) STOP --OPEN ERROR (BGD) --
C
C       INPUT IMAGE PARAMETERS
C
        ACCEPT"IMAGE SIZE (NX,NY) = ",NX,NY
        TYPE"COORDS OF UPPER LH CORNER OF IMAGE"
        TYPE"E.G. (1,1)= UPPER LH CORNER OF ENTIRE ARRAY"
        ACCEPT"(NXC,NYC) = ",NXC,NYC
        JCMX=NX+NXC-1
        JRMX=NY+NYC-1
        IF(JCMX.GT.256) STOP--IMAGE OFF RIGHT EDGE OF ARRAY--

        PI=3.1415926
        ARG=PI/8.
        DO 10 I=1,256
10      IDD(I)=0
        IDD(2)=256
        IDD(4)=JRMX
        CALL WRBLK(1,0,IDD,1,IER)
        IF(IER.NE.1)STOP--WRBLK ERR--
        IDD(2)=0
        IDD(4)=0
        NS1=NYC-1
        IF(NS1.LE.0) GO TO 30
        DO 20 IR=1,NS1
20      CALL WRBLK(1,IR,IDD,1,IER)
        IF(IER.NE.1) STOP--WRBLK ERR--
30      CONTINUE
        DO 50 IR=NYC,JRMX
        IIR=IR+NYC-1
        IF(IIR.LT.29.OR.IIR.GT.35)GO TO 555
        DO 5 IC=1,NX
        IIC=IC+NXC-1
        IDD(32)=100
        IF(IIR.EQ.32.AND.IIC.EQ.30)IDD(IIC)=100
        IF(IIR.EQ.32.AND.IIC.EQ.31)IDD(IIC)=100
        IF(IIR.EQ.32.AND.IIC.EQ.33)IDD(IIC)=100
        IF(IIR.EQ.32.AND.IIC.EQ.34)IDD(IIC)=100
        IF(IIR.EQ.32.AND.IIC.EQ.29)IDD(IIC)=100
        IF(IIR.EQ.32.AND.IIC.EQ.35)IDD(IIC)=100
C
        IF(IIR.EQ.29.AND.IIC.EQ.49)IDD(IIC)=100
        IF(IIR.EQ.30.AND.IIC.EQ.49)IDD(IIC)=100
        IF(IIR.EQ.31.AND.IIC.EQ.49)IDD(IIC)=100
        IF(IIR.EQ.32.AND.IIC.EQ.49)IDD(IIC)=100
        IF(IIR.EQ.32.AND.IIC.EQ.50)IDD(IIC)=100
        IDD(51)=100
C
  554   CONTINUE
5       CONTINUE
  555   CONTINUE
        CALL WRBLK(1,IR,IDD,1,IER)
        IF(IER.NE.1) STOP --WRBLK ERROR (BGD)--
        DO 6 I6=1,NX
        IDD(I6)=0
    6   CONTINUE
  556   CONTINUE
50      CONTINUE

        CALL CLOSE(1,IER)
   80   CONTINUE
        STOP
        END
```

```
C    CR5              12/20/78
C
C         CRSCN GENERATES AN 256 X JRMX ARRAY CONTAINING
C         AN NX X NY IMAGE WITH UPPER LH CORNER COORDS (NXC,NYC)
C         THE PORTIONS OF THE ARRAY NOT CONTAINING THE IMAGE ARE
C         FILLED WITH ZEROS AND THE ARRAY IS STORED ON DISK AS KNAME
C
          DIMENSION IDD(256),KNAME(7)

C
C         KNAME IN FILE #1 IS USED TO STORE THE BACKGROUND IMAGE

          RMEAN=0
          SIGMA=256
          NRAND=2351
          ACCEPT "RANDOM SEED",NRAND
          DO 80 I80=1,10
          TYPE "RUN ",I80
          ACCEPT "REFERENCE SIZE = ",NSIZE
          ACCEPT "REFERENCE POSITION = ",NPOS
          WRITE(10,900)
900       FORMAT(" NAME OF FILE TO BE CREATED = ",Z)
          READ(11,905) KNAME(1)
905       FORMAT(S7)
          CALL DFILW(KNAME(1),IER)
          CALL CFILW(KNAME(1),2,IER)
          IF(IER.NE.1) STOP -- CFILW ERROR (BGD) --
          CALL OPEN(1,KNAME(1),2,IER,512)
          IF(IER.NE.1) STOP --OPEN ERROR (BGD) --
C
C         INPUT IMAGE PARAMETERS
C
          ACCEPT"IMAGE SIZE (NX,NY) = ",NX,NY
          TYPE"COORDS OF UPPER LH CORNER OF IMAGE"
          TYPE"E.G.  (1,1)= UPPER LH CORNER OF ENTIRE ARRAY"
          ACCEPT"(NXC,NYC) = ",NXC,NYC
          JCMX=NX+NXC-1
          JRMX=NY+NYC-1
          IF(JCMX.GT.256) STOP--IMAGE OFF RIGHT EDGE OF ARRAY--

          PI=3.1415926
          ARG=PI/8
          DO 10 I=1,256
10        IDD(I)=0
          IDD(2)=256
          IDD(4)=JRMX
          CALL WRBLK(1,0,IDD,1,IER)
          IF(IER.NE.1)STOP--WRBLK ERR--
          IDD(2)=0
          IDD(4)=0
          NS1=NYC-1
          IF(NS1.LE.0) GO TO 30
          DO 20 IR=1,NS1
20        CALL WRBLK(1,IR,IDD,1,IER)
          IF(IER.NE.1) STOP--WRBLK ERR--
30        CONTINUE
          DO 50 IR=NYC,JRMX
          IIR=IR+NYC-1
          IF(IIR.LT.29.OR.IIR.GT.35)GO TO 555
          DO 5 IC=1,NX
          IIC=IC+NXC-1
          IDD(32)=100
          IF(IIR.EQ.32.AND.IIC.EQ.30)IDD(IIC)=100
          IF(IIR.EQ.32.AND.IIC.EQ.31)IDD(IIC)=100
          IF(IIR.EQ.32.AND.IIC.EQ.33)IDD(IIC)=100
          IF(IIR.EQ.32.AND.IIC.EQ.34)IDD(IIC)=100
          IF(IIR.EQ.32.AND.IIC.EQ.29)IDD(IIC)=100
          IF(IIR.EQ.32.AND.IIC.EQ.35)IDD(IIC)=100
C
          IF(IIR.EQ.29.AND.IIC.EQ.49)IDD(IIC)=100
          IF(IIR.EQ.29.AND.IIC.EQ.50)IDD(IIC)=100
          IF(IIR.EQ.29.AND.IIC.EQ.51)IDD(IIC)=100
          IF(IIR.EQ.30.AND.IIC.EQ.49)IDD(IIC)=100
          IF(IIR.EQ.31.AND.IIC.EQ.49)IDD(IIC)=100
          IF(IIR.EQ.32.AND.IIC.EQ.49)IDD(IIC)=100
          IF(IIR.EQ.32.AND.IIC.EQ.50)IDD(IIC)=100
          IF(IIR.EQ.32.AND.IIC.EQ.51)IDD(IIC)=100
          IF(IIR.EQ.33.AND.IIC.EQ.51)IDD(IIC)=100
          IF(IIR.EQ.34.AND.IIC.EQ.51)IDD(IIC)=100
          IF(IIR.EQ.35.AND.IIC.EQ.49)IDD(IIC)=100
          IF(IIR.EQ.35.AND.IIC.EQ.50)IDD(IIC)=100
          IF(IIR.EQ.35.AND.IIC.EQ.51)IDD(IIC)=100
C
  554     CONTINUE
5         CONTINUE
  555     CONTINUE
          CALL WRBLK(1,IR,IDD,1,IER)
          IF(IER.NE.1) STOP --WRBLK ERROR (BGD)--
          DO 6 I6=1,NX
          IDD(I6)=0
    6     CONTINUE
  556     CONTINUE
50        CONTINUE

          CALL CLOSE(1,IER)
   80     CONTINUE
          STOP
          END
```

```
C    CR6              12/20/78
C
C         CRSCN GENERATES AN 256 X JRMX ARRAY CONTAINING
C         AN NX X NY IMAGE WITH UPPER LH CORNER COORDS (NXC,NYC).
C         THE PORTIONS OF THE ARRAY NOT CONTAINING THE IMAGE ARE
C         FILLED WITH ZEROS AND THE ARRAY IS STORED ON DISK AS KNAME
C
          DIMENSION IDD(256),KNAME(7)

C
C         KNAME IN FILE #1 IS USED TO STORE THE BACKGROUND IMAGE

          RMEAN=0.
          SIGMA=256.
          NRAND=2351
          ACCEPT "RANDOM SEED",NRAND
          DO 80 I80=1,10
          TYPE "RUN ",I80
          ACCEPT "REFERENCE SIZE = ",NSIZE
          ACCEPT "REFERENCE POSITION = ",NPOS
          WRITE(10,900)
900       FORMAT(" NAME OF FILE TO BE CREATED = ",Z)
          READ(11,905) KNAME(1)
905       FORMAT(S7)
          CALL DFILW(KNAME(1),IER)
          CALL CFILW(KNAME(1),2,IER)
          IF(IER.NE.1) STOP -- CFILW ERROR (BGD) --
          CALL OPEN(1,KNAME(1),2,IER,512)
          IF(IER.NE.1) STOP --OPEN ERROR (BGD) --
C
C         INPUT IMAGE PARAMETERS
C
          ACCEPT"IMAGE SIZE (NX,NY) = ",NX,NY
          TYPE"COORDS OF UPPER LH CORNER OF IMAGE"
          TYPE"E.G.  (1,1)= UPPER LH CORNER OF ENTIRE ARRAY"
          ACCEPT"(NXC,NYC) = ",NXC,NYC
          JCMX=NX+NXC-1
          JRMX=NY+NYC-1
          IF(JCMX .GT. 256) STOP--IMAGE OFF RIGHT EDGE OF ARRAY--

          PI=3.1415926
          ARG=PI/8.
          DO 10 I=1,256
10        IDD(I)=0
          IDD(2)=256
          IDD(4)=JRMX
          CALL WRBLK(1,0,IDD,1,IER)
          IF(IER .NE. 1)STOP--WRBLK ERR--
          IDD(2)=0
          IDD(4)=0
          NS1=NYC-1
          IF(NS1 .LE. 0) GO TO 30
          DO 20 IR=1,NS1
20        CALL WRBLK(1,IR,IDD,1,IER)
          IF(IER .NE. 1) STOP--WRBLK ERR--
30        CONTINUE
          DO 50 IR=NYC,JRMX
          IIR=IR+NYC-1
          IF(IIR .LT. 29 .OR. IIR .GT. 35)GO TO 555
          DO 5 IC=1,NX
          IIC=IC+NXC-1
          IDD(32)=100
          IF(IIR .EQ. 32 .AND. IIC .EQ. 30)IDD(IIC)=100
          IF(IIR .EQ. 32 .AND. IIC .EQ. 31)IDD(IIC)=100
          IF(IIR .EQ. 32 .AND. IIC .EQ. 33)IDD(IIC)=100
          IF(IIR .EQ. 32 .AND. IIC .EQ. 34)IDD(IIC)=100
          IF(IIR .EQ. 32 .AND. IIC .EQ. 29)IDD(IIC)=100
          IF(IIR .EQ. 32 .AND. IIC .EQ. 35)IDD(IIC)=100
C
          IF(IIR .EQ. 29 .AND. IIC .EQ. 49)IDD(IIC)=100
          IF(IIR .EQ. 29 .AND. IIC .EQ. 50)IDD(IIC)=100
          IF(IIR .EQ. 29 .AND. IIC .EQ. 51)IDD(IIC)=100
          IF(IIR .EQ. 30 .AND. IIC .EQ. 49)IDD(IIC)=100
          IF(IIR .EQ. 31 .AND. IIC .EQ. 49)IDD(IIC)=100
          IF(IIR .EQ. 32 .AND. IIC .EQ. 49)IDD(IIC)=100
          IF(IIR .EQ. 32 .AND. IIC .EQ. 50)IDD(IIC)=100
          IF(IIR .EQ. 32 .AND. IIC .EQ. 51)IDD(IIC)=100
          IF(IIR .EQ. 33 .AND. IIC .EQ. 51)IDD(IIC)=100
          IF(IIR .EQ. 33 .AND. IIC .EQ. 49)IDD(IIC)=100
          IF(IIR .EQ. 34 .AND. IIC .EQ. 51)IDD(IIC)=100
          IF(IIR .EQ. 34 .AND. IIC .EQ. 49)IDD(IIC)=100
          IF(IIR .EQ. 35 .AND. IIC .EQ. 49)IDD(IIC)=100
          IF(IIR .EQ. 35 .AND. IIC .EQ. 50)IDD(IIC)=100
          IF(IIR .EQ. 35 .AND. IIC .EQ. 51)IDD(IIC)=100
C
  554     CONTINUE
5         CONTINUE
  555     CONTINUE
          CALL WRBLK(1,IR,IDD,1,IER)
          IF(IER.NE.1) STOP --WRBLK ERROR (BGD)--
          DO 6 I6=1,NX
          IDD(I6)=0
    6     CONTINUE
  556     CONTINUE
50        CONTINUE

          CALL CLOSE(1,IER)
   80     CONTINUE
          STOP
          END
```

```
C   CRBG.FR          11/10/78
C
C       CRBGN GENERATES AN 256 X JRMX ARRAY CONTAINING
C       AN NX X NY IMAGE WITH UPPER LH CORNER COORDS (NXC,NYC)
C       THE PORTIONS OF THE ARRAY NOT CONTAINING THE IMAGE ARE
C       FILLED WITH ZEROS AND THE ARRAY IS STORED ON DISK AS KNAME
C
        DIMENSION IDD(256),KNAME(7)

C
C       KNAME IN FILE #1 IS USED TO STORE THE BACKGROUND IMAGE

        RMEAN=0
        SIGMA=256
        NRAND=2351
        ACCEPT "RANDOM SEED",NRAND
        DO 80 I80=1,10
        TYPE "RUN ",I80
        ACCEPT "REFERENCE SIZE = ",NSIZE,ISIZE
        ACCEPT "REFERENCE POSITION = ",NPOS
        WRITE(10,900)
900     FORMAT(" NAME OF FILE TO BE CREATED = ",Z)
        READ(11,905) KNAME(1)
905     FORMAT(S7)
        CALL DFILW(KNAME(1),IER)
        CALL CFILW(KNAME(1),2,IER)
        IF(IER.NE.1) STOP -- CFILW ERROR (BGD) --
        CALL OPEN(1,KNAME(1),2,IER,512)
        IF(IER.NE.1) STOP --OPEN ERROR (BGD) --
C
C       INPUT IMAGE PARAMETERS
C
        ACCEPT"IMAGE SIZE (NX,NY) = ",NX,NY
        TYPE"COORDS OF UPPER LH CORNER OF IMAGE"
        TYPE"E.G. (1,1)= UPPER LH CORNER OF ENTIRE ARRAY"
        ACCEPT"(NXC,NYC) = ",NXC,NYC
        JCMX=NX+NXC-1
        JRMX=NY+NYC-1
        IF(JCMX.GT.256) STOP--IMAGE OFF RIGHT EDGE OF ARRAY--

        PI=3.1415926
        ARG=PI/8.
        DO 10 I=1,256
10      IDD(I)=0
        IDD(2)=256
        IDD(4)=JRMX
        CALL WRBLK(1,0,IDD,1,IER)
        IF(IER.NE.1)STOP--WRBLK ERR--
        IDD(2)=0
        IDD(4)=0
        NS1=NYC-1
        IF(NS1.LE.0) GO TO 30
        DO 20 IR=1,NS1
20      CALL WRBLK(1,IR,IDD,1,IER)
        IF(IER.NE.1) STOP--WRBLK ERR--
30      CONTINUE
        DO 50 IR=NYC,JRMX
        IIR=IR+NYC-1
        IF(IIR.LT.NPOS.OR.IIR.GE.NPOS+NSIZE)GO TO 555
        DO 5 IC=1,NX
        IIC=IC+NXC-1
  300   IF(IIC.LT.NPOS.OR.IIC.GE.NPOS+ISIZE)GO TO 554
        IDD(IIC)=100
  554   CONTINUE
5       CONTINUE
  555   CONTINUE
        CALL WRBLK(1,IR,IDD,1,IER)
        IF(IER.NE.1) STOP --WRBLK ERROR (BGD)--
        IF(IIR.LT.NPOS.OR.IIR.GE.NPOS+NSIZE)GO TO 556
        DO 6 I6=1,NX
        IDD(I6)=0
    6   CONTINUE
  556   CONTINUE
50      CONTINUE

        CALL CLOSE(1,IER)
   80   CONTINUE
        STOP
        END
```

```
C IMAGER.FR      11/3/78
C
C         THIS ROUTINE READS AN INTEGER IMAGE
C         AND OUTPUTS A GRAYSCALE PLOT
C
          COMPLEX IS(64,64)
          DIMENSION KNAME(7)
          NR=64
          NC=64
          M=6
          IAT=1
          IMN=1
          CALL SDEF(IS,M,IMN,IAT,KNAME(1))
          CALL GSPLT(IS,NC,NR,3)
          STOP
          END
```

```
IREAD.FR      09/01/78
C
      SUBROUTINE SDEF(IS,M,IMN,IAT,KNAME)
C
C     SDEF READS A PORTION OF A REAL IMAGE STORED ON DISK
C     AND STORES IT AS THE REAL COMPONENT OF COMPLEX ARRAY IS
C
      DIMENSION KI(256,5),KNAME(7)
      COMPLEX IS(64,64),CZERO
C
      IMN=IMN+1
C
      NC=2**M
      NR=NC
      MM=M
      NCC=2**MM
      NRR=NCC
C
      WRITE(10,900)
900   FORMAT(" INPUT FILENAME = ",Z)
      READ(11,905) KNAME(1)
905   FORMAT(S7)
      CALL OPEN(1,KNAME(1),0,IER,512)
      IF(IER .NE. 1)STOP--OPEN ERR--
910   FORMAT(" INPUT FILENAME: ",S10)
      CZERO=CMPLX(0.,0.)
      DO 17 I5=1,NR
      DO 17 I6=1,NC
17    IS(I6,I5)=CZERO
      CALL RDBLK(1,0,KI,1,IER)
      IF(IER .NE. 1)STOP--RDBL ERR--
      NCI=KI(2,1)
      NRI=KI(4,1)
      TYPE "NCI,NRI",NCI,NRI
      TYPE"INPUT (X,Y) COORDS OF UPPER LH CORNER OF IMAGE"
      TYPE"E.G. (1,1)= UPPER LH CORNER OF IMAGE ON DISK"
      ACCEPT"     = ",NXC,NYC
920   FORMAT(" UPPER LH COORDS (X,Y) =",2I5)
      JCMX=NXC+NCC-1
      JRMX=NYC+NRR-1
      IF(JCMX .GT. NCI) STOP--NXC TOO LARGE
      IF(JRMX .GT. NRI) STOP--NYC TOO LARGE
      II=1
      IR=-4
      NBLOCK=5
      LOOP=NRR/NBLOCK
      LOB=NRR-(LOOP*NBLOCK)
      IF(LOOP .EQ. 0)GO TO 1001
      DO 40 I40=1,LOOP
      IR=(I40-1)*5+NYC
      CALL RDBLK(1,IR,KI,NBLOCK,IER)
      IF(IER .NE. 1)STOP--RDBL ERR--
1002  CONTINUE
      JJ=0
      DO 42 I42=1,NCC
      IC=I42+NXC-1
      JJ=JJ+1
      DO 43 I43=1,NBLOCK
      II43=I43-1
      SS=FLOAT(KI(IC,I43))
      IF(SS .GT. 1023)SS=1023.
      IS(JJ,II+II43)=CMPLX(SS,0.)
43    CONTINUE
42    CONTINUE
      II=II+5
1001  CONTINUE
40    CONTINUE
      IF(LOB .EQ. 0)GO TO 50
      NBLOCK=LOB
      LOB=0
      IR=IR+5
      CALL RDBLK(1,IR,KI,NBLOCK,IER)
      IF(IER .NE. 1)STOP --RDBL ERR-
      GO TO 1002
50    CONTINUE
      TYPE IS(1,1),IS(64,64),IS(10,2),IS(2,10)
      CALL CLOSE(1,IER)
      RETURN
      END
```

4.2.2 HDRV.FR

HDRV.FR is a FORTRAN program which creates a single hologram and performs an image reconstruction. The holograph is created by multiplying the two-dimensional Fourier transform of an image by the complex conjugate of the Fourier transform of a reference image. The multiplication is done on a point-to-point basis. The reconstruction is accomplished by multiplying the hologram by the Fourier transform of the reference on a point-by-point basis, and taking the inverse Fourier transform. The reconstructed image is written to a disk file, in complex format.

```
C HDRV.FR        10/20/78          REV. A   10/31/78
C
C       THIS IS A DRIVER PROGRAM TO GENERATE HOLOGRAM
C       DATA FILES
C
C
        DIMENSION KNAME(7),JREF(7),KSUM(7),KFNAME(7),JFREF(7)
        COMPLEX IS(64,64),TEMP(64)
        IMN=1
        M=6
        IAT=1
        NR=2**M
        NC=NR
        TYPE "SUBJECT INPUT"
        CALL SDEF(IS,M,IMN,IAT,KNAME(1))
        CALL FFT2D(IS,NC,NR,M,1)
        CALL WBLKC(IS,KFNAME(1))
        TYPE "REFERENCE INPUT"
        CALL SDEF(IS,M,IMN,IAT,JREF(1))
        CALL FFT2D(IS,NC,NR,M,1)
        CALL WBLKC(IS,JFREF(1))
C
C       OPEN FILE OF COMPLEX IMAGE FREQUENCIES TO BE READ
C
900     FORMAT(" NAME OF FILE TO BE READ = ",Z)
905     FORMAT(S7)
        CALL OPEN(1,KFNAME(1),2,IER,512)
        IF(IER .NE. 1)STOP --OPEN ERROR--
C
C       CONJUGATE REFERENCE AND PERFORM POINT TO POINT MULTIPLY
C
        NBLOCK=1
        DO 40 I40=1,64
        IR=I40
        CALL RDBLK(1,IR,TEMP,NBLOCK,IER)
        IF(IER .NE. 1)STOP --RDBLK ERROR--
  999   FORMAT(1X,8E14.4)
        DO 40 I41=1,64
        IS(I41,I40)=CONJG(IS(I41,I40))*TEMP(I41)
40      CONTINUE
C
C       CLOSE IMAGE FILE
C
        CALL CLOSE(1,IER)
        IF(IER .NE. 1)STOP --CLOSE ERROR--
C
C       OPEN REFERENCE FILE
C
        CALL OPEN(1,JFREF(1),2,IER,512)
        IF(IER .NE. 1)STOP --OPEN ERROR--
C
C       PERFORM RECONSTRUCTION
C
        DO 50 I50=1,64
        IR=I50
        CALL RDBLK(1,IR,TEMP,NBLOCK,IER)
        DO 50 I51=1,64
        IS(I51,I50)=IS(I51,I50)*TEMP(I51)
50      CONTINUE
C
C       PERFORM INVERSE FFT
C
        CALL FFT2D(IS,NC,NR,M,2)
C
C       STORE RECONSTRUCTED IMAGE
C
        CALL CLOSE(1,IER)
        IF(IER .NE. 1)STOP --CLOSE ERROR--
        TYPE "HOLOGRAM OUTPUT"
        CALL WBLKC(IS,KSUM(1))
        STOP
        END
```

```
C HDRV.FR        10/20/78
C
C       THIS IS A DRIVER PROGRAM TO GENERATE HOLOGRAM
C       DATA FILES
C
C
        DIMENSION KNAME(7)
        COMPLEX CVALUE
        COMPLEX IS(64,64),TEMP(64)
        IMN=1
        M=6
        IAT=1
        NR=2**M
        NC=NR
        ACCEPT "VALUE = ",VALUE
        CVALUE=CMPLX(VALUE,0.)
        CALL SDEF(IS,M,IMN,IAT)
        CALL WBLKC(IS)
        CALL FFT2D(IS,NC,NR,M,1)
        DO 22 I=1,64
        DO 22 J=1,64
        IS(I,J)=CVALUE*IS(I,J)
22      CONTINUE
        CALL FFT2D(IS,NC,NR,M,2)
        CALL WBLKC(IS)
        STOP
        CALL SDEF(IS,M,IMN,IAT)
        CALL FFT2D(IS,NC,NR,M,1)
        CALL WBLKC(IS)
C
C       OPEN FILE OF COMPLEX IMAGE FREQUENCIES TO BE READ
C
        WRITE(10,900)
900     FORMAT(" NAME OF FILE TO BE READ = ",Z)
        READ(11,905)KNAME(1)
905     FORMAT(S7)
        CALL OPEN(1,KNAME(1),2,IER,512)
        IF(IER .NE. 1)STOP --OPEN ERROR--
C
C       CONJUGATE REFERENCE AND PERFORM POINT TO POINT MULTIPLY
C
        NBLOCK=1
        DO 40 I40=1,64
        IR=I40
        CALL RDBLK(1,IR,TEMP,NBLOCK,IER)
        IF(IER .NE. 1)STOP --RDBLK ERROR--
  999   FORMAT(1X,8E14.4)
        DO 40 I41=1,64
        IS(I41,I40)=CONJG(IS(I41,I40))*TEMP(I41)
40      CONTINUE
C
C       CLOSE IMAGE FILE
C
        CALL CLOSE(1,IER)
        IF(IER .NE. 1)STOP --CLOSE ERROR--
C
C       OPEN REFERENCE FILE
C
        WRITE(10,900)
        READ(11,905)KNAME(1)
        CALL OPEN(1,KNAME(1),2,IER,512)
        IF(IER .NE. 1)STOP --OPEN ERROR--
C
C       PERFORM RECONSTRUCTION
C
        DO 50 I50=1,64
        IR=I50
        CALL RDBLK(1,IR,TEMP,NBLOCK,IER)
        DO 50 I51=1,64
        IS(I51,I50)=IS(I51,I50)*TEMP(I51)
50      CONTINUE
C
C       PERFORM INVERSE FFT
C
        CALL FFT2D(IS,NC,NR,M,2)
C
C       STORE RECONSTRUCTED IMAGE
C
        CALL CLOSE(1,IER)
        IF(IER .NE. 1)STOP --CLOSE ERROR--
        CALL WBLKC(IS)
        STOP
        END
```

4.2.3 HSDRV.FR

HSDRV.FR is a FORTRAN program used to create a multiplexed hologram. The program accepts image and reference integer files, computes the hologram, and adds the result to a specified hologram file. The program requires a hologram file; therefore, CRSUM.SV is usually run prior to running HSDRV.SV to create a file in the proper format filled with zeros. The images are reconstructed using the MHOLO.SV program.

The operator and program interface via the console is similar in terms of file inputs to the FBDRVT.FR program. Refer to FBDRVT.FR operator/program dialogue (Paragraph 4.2.5) for file formats.

4.2.4 MHOLO.FR

MHOLO.FR is a FORTRAN program which reads a specified hologram file and performs an image reconstruction and gray scale plot for a specified reference file. The reference file must contain the Fourier Transform of the reference image.

The operator and program dialogue uses the standard machine holograph nomenclature detail in the FBDRVT.FR description (Paragraph 4.2.5).

```
C HSDRV.FR      10/31/78        REV. A  11/3/78
C
C       THIS IS A DRIVER PROGRAM TO GENERATE HOLOGRAM
C       DATA FILES
C
C
        DIMENSION KNAME(7),JREF(7),KSUM(7),KFNAME(7),JFREF(7)
        COMPLEX IS(64,64),TEMP(64),CVALUE
        CVALUE=CMPLX(1.,0.)
        ACCEPT "WEIGHTING COEFFICIENT ",WC
        CVALUE=CMPLX(WC,0.)
        WRITE(10,900)
        READ(11,905)KSUM(1)
        IMN=1
        M=6
        IAT=1
        NR=2**M
        NC=NR
        ACCEPT "NUMBER OF RUNS ",NRUNS
        DO 80 I80=1,NRUNS
        TYPE "RUN ",I80
        TYPE "SUBJECT INPUT"
        CALL SDEF(IS,M,IMN,IAT,KNAME(1))
        CALL FFT2D(IS,NC,NR,M,1)
        CALL WBLKC(IS,KFNAME(1))
        TYPE "RUN ",I80
        TYPE "REFERNCE INPUT "
        CALL SDEF(IS,M,IMN,IAT,JREF(1))
        CALL FFT2D(IS,NC,NR,M,1)
        CALL WBLKC(IS,JFREF(1))
C
C       OPEN FILE OF COMPLEX IMAGE FREQUENCIES TO BE READ
C
900     FORMAT(" NAME OF FILE TO BE READ = ",Z)
905     FORMAT(S7)
        CALL OPEN(1,KFNAME(1),2,IER,512)
        IF(IER .NE. 1)STOP --OPEN ERROR--
C
C       CONJUGATE REFERENCE AND PERFORM POINT TO POINT MULTIPLY
C
        NBLOCK=1
        DO 40 I40=1,64
        IR=I40
        CALL RDBLK(1,IR,TEMP,NBLOCK,IER)
        IF(IER .NE. 1)STOP --RDBLK ERROR--
  999   FORMAT(1X,8E14.4)
        DO 40 I41=1,64
        IS(I41,I40)=CONJG(IS(I41,I40))*TEMP(I41)
40      CONTINUE
C
C       CLOSE IMAGE FILE
C
        CALL CLOSE(1,IER)
        IF(IER .NE. 1)STOP --CLOSE ERROR--
C
C       OPEN HOLOGRAM SUM FILE
C
        CALL OPEN(1,KSUM(1),2,IER,512)
        IF(IER .NE. 1)STOP --OPEN ERROR--
C
C       PERFORM SUMMATION
C
        DO 50 I50=1,64
        IR=I50
        CALL RDBLK(1,IR,TEMP,NBLOCK,IER)
        DO 50 I51=1,64
        IS(I51,I50)=CVALUE*IS(I51,I50)+TEMP(I51)
50      CONTINUE
C
C       STORE HOLOGRAM SUM
C
        CALL CLOSE(1,IER)
        IF(IER .NE. 1)STOP --CLOSE ERROR--
        CALL WBLKC(IS,KSUM(1))
   80   CONTINUE
        STOP
        END
```

```
C MHOLO.FR       11/3/78
C
C        THIS PROGRAM READS A MULTIPLEXED HOLOGRAM
C        FROM DISK, PERFORMS IMAGE RECONSTRUCTION
C        AND OUTPUTS A GRAYSCALE PLOT
C
C
         DIMENSION KNAME(7),KSUM(7)
         COMPLEX IS(64,64),TEMP(64)
         M=6
         NC=64
         NR=64
C
C
C        READ FILE CONTAINING MULTIPLEXED HOLOGRAM
C
C
         NBLOCK=1
         ACCEPT "NUMBER OF RUNS ",NRUNS
         DO 80 I80=1,NRUNS
         TYPE "RUN ",I80
         TYPE "HOLOGRAM FILE"
         CALL RBLKC(IS)
C
C        INPUT REFERENCE IMAGE TRANSFORM (F-DOMAIN)
C
         WRITE(10,900)
  900    FORMAT(" NAME OF REFERENCE IS ",Z)
         READ(11,905)KNAME(1)
  905    FORMAT(S7)
         CALL OPEN(1,KNAME(1),2,IER,512)
         IF(IER .NE. 1)STOP --OPEN ERROR--
C
C        AND DO RECONSTRUCTION
C
         DO 50 I50=1,64
         IR=I50
         CALL RDBLK(1,IR,TEMP,NBLOCK,IER)
         DO 50 I51=1,64
         IS(I51,I50)=IS(I51,I50)*TEMP(I51)
   50    CONTINUE
C
C        PERFORM INVERSE TRANSFORM
C
         CALL FFT2D(IS,NC,NR,M,2)
C
         CALL CLOSE(1,IER)
         IF(IER .NE. 1)STOP --CLOSE ERROR--
         CALL GSPLT(IS,NC,NR,3)
         WRITE(12,907)KNAME(1)
  907    FORMAT(1H0,S7)
   80    CONTINUE
         STOP
         END
```

4.2.5 FBDRVT

FBDRVT.FR is a FORTRAN driver program for hologram feedback systems. FBDRVT requires the following relocatable binary files:

WBLC.RB

IREAD.RB

FFT2D.RB

FNORM.RB

CONS.RB

RBLC.RB

HCRE.RB

HRCON.RB

HADD.RB

GSPLT.RB

PICPLT.RB

PLPT.RB

FORT.LB

The proper load statement is contained in file FBDRVT.MC

4.2.6 FBDRVT.FR

FBDRVT.FR is a FORTRAN driver program used to examine multiplexed holograms in feedback systems and is illustrated in Figures 18 and 19. The program performs the equation

$$H_n = \frac{H_{n-1} + c\ R^* S_n}{(1 + c\ RR^*)} \tag{59}$$

where

H_n = hologram in Fourier domain at iteration n

R = reference image in Fourier domain

R^* = complex conjugate of R

S_n = image scene in Fourier domain

c = convergence constant

for operator specified scene and reference image pairs.

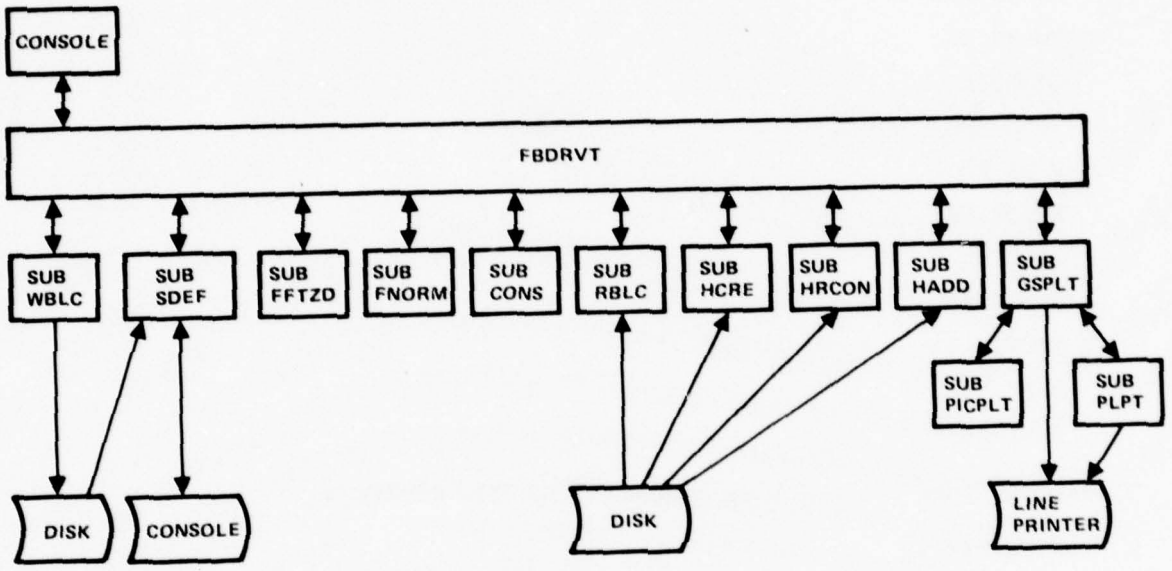


FIGURE 18. FBDRVT SUBROUTINE INTERFACE DIAGRAM

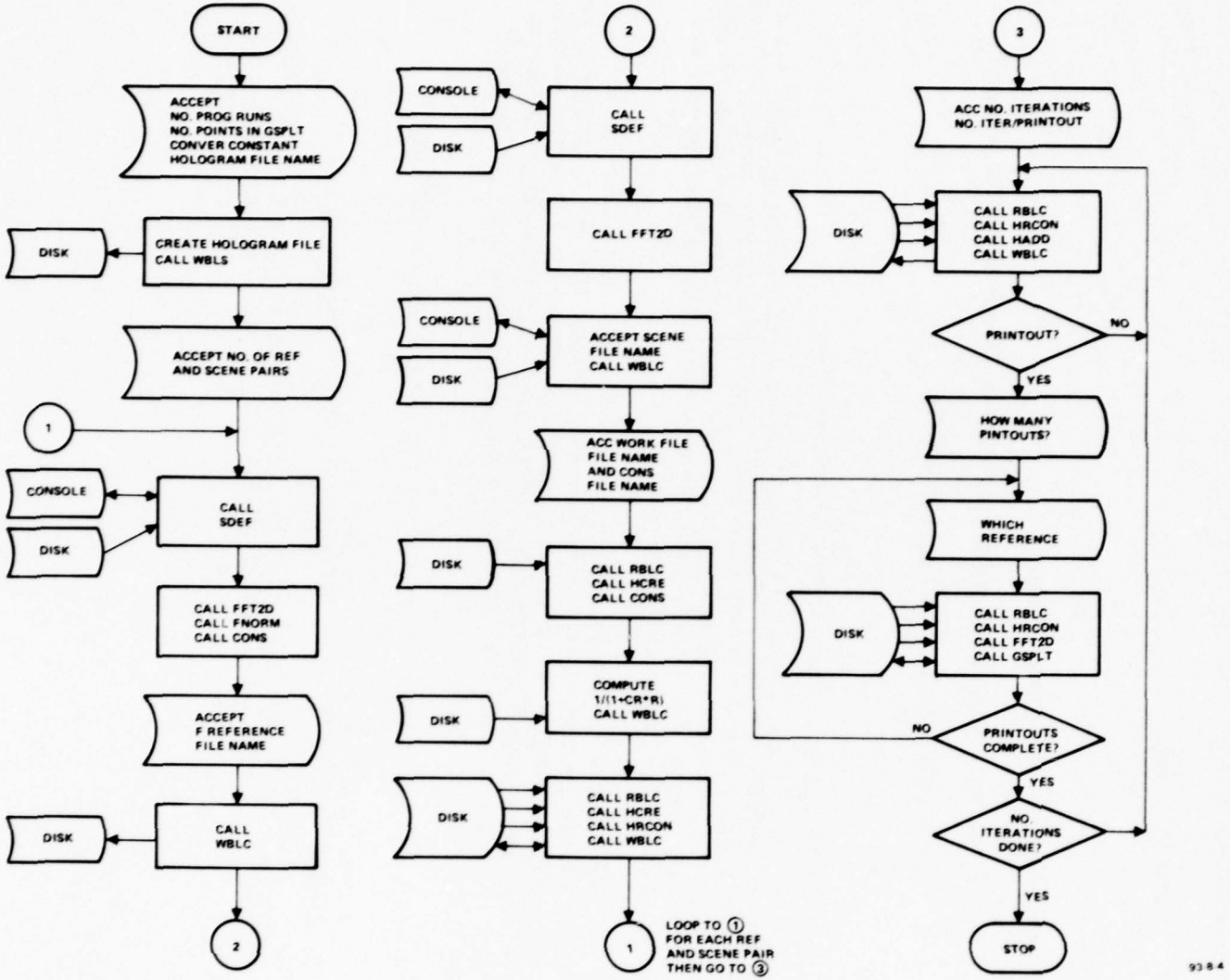


FIGURE 19. PSEUDO FLOWCHART FOR FBDRVT

Reconstruction of the image is accomplished by using the formula

$$I_A = F^{-1} (R_A \cdot H) \tag{60}$$

where

I_A = the reconstructed image

R_A = the reference in the Fourier domain paired with the image I_A

H = the hologram in the Fourier domain

and

F^{-1} = the inverse Fourier transform operator

The driver program is functionally divided into two divisions. The first division sets up the scene and reference data files through console interface with the operator. The second division performs the hologram feedback equation (Equation 59) and the reconstruction equation (Equation 60) for operator specified image and scene pairs.

In the first division the program asks the operator for files containing integer reference images and integer scene images. For each reference and scene pair the program will ask for four file names for files the program creates. The second division the program performs the feedback equation and image reconstruction. The output is a grayscale plot for each operator specified reconstruction.

The dialogue between the operator and the program is as follows:

NUMBER OF PROGRAM RUNS - integer
NUMBER OF POINTS TO BE AVERAGED IN GSPLT - integer

(NOTE: The grayscale plot finds the specified number of peak (highest) values and averages them to calculate S/N (see GSPLT. description).)

CONVERGENCE CONSTANT = realnumber
NAME OF HOLOGRAM FILE TO BE CREATED = filename
NUMBER OF REFERENCES AND SCENE PAIRS = integer

(NOTE: The program can perform the feedback equation on a maximum of six reference and scene pairs, and the hologram reconstruction equation on a maximum of nine references. The maximum integer response to the message above is nine.)

The following set of responses is repeated for the specified number of reference and scene pairs.

REFERENCE INPUT
INPUT FILENAME = filename

(NOTE: The specified file must be an integer image in the standard format (see CRSCN description).)

NCI,NRI 64,64

(NOTE: The number of columns and the number of rows of all hologram images are 64 by 64.)

INPUT (X,Y) COORDS OF UPPER LH CORNER OF IMAGE
E.G. (1,1) = UPPER LH CORNER OF IMAGE ON DISK = 1,1

(NOTE: All hologram images have coordinates 1,1 for the upper left hand corner of the image.)

IN FFT2D
OUT FFT2D

indicates that Fourier transform has been performed.

NAME OF REF FILE IN F DOMAIN = filename

(NOTE: The program deletes the specified file and creates a new file containing the 64 by 64 complex array of the reference Fourier transform. In the past it has been the practice to start file names of transformed images with the letter F to indicate the data is in the Fourier domain.)

SCENE INPUT
INPUT FILENAME = filename

(NOTE: The specified file must be an integer scene in the standards format (see CRSCN description).)

NCI,NRI 64,64
INPUT (X,Y) COORDS OF UPPER LH CORNER OF IMAGE
E.G. (1,1) = UPPER LH CORNER OF IMAGE ON DISK = 1,1
IN FFT2D
OUT FFT2D
NAME OF IMAGE FILE IN F DOMAIN - filename

(NOTE: The program deletes the specified file and creates a new file containing the 64 by 64 complex array of the scene Fourier transform. In the past it has been the practice to start file names of Fourier transformed files with the letter F to indicate the data is in the F domain.)

NAME OF WORKING FILE = filename

(NOTE: The program creates a file containing $1/(1 + c\ RR^*)$ (see Equation 59) for each reference and scene pair. This is called the working file. The file name must be different for each reference and scene pair. If a reference is to be used for reconstruction only, a file name for the working file still must be specified for each reference and scene pair.)

NAME OF CONSTANT FILE = filename

(NOTE: The program creates a file containing

$$\frac{c\ R^*S_n}{(1 + c\ RR^*}$$

for each reference and scene pair (see Equation 59). This is called the constant file. A different file name must be specified for each reference and scene pair. If a reference is used reconstruction only, a file name for the constant file still must be specified for each reference and scene pair.)

This ends the file setup division of the program.

The following is the dialogue between the operator and the program for the feedback and image reconstruction section of the program.

NUMBER OF ITERATIONS = integer

(NOTE: The program will perform Equation 59 the specified number of times and then stop, or return to the beginning of the program.)

NUMBER OF ITERATIONS PER PRINTOUT = integer

(NOTE: The first time through the loop the program computes Equation 59 only once prior to a grayscale plot.)

The following message is repeated every iteration until a printout.

WHICH REFERENCE AND SCENE = integer

(NOTE: The integer response indicates which reference and scene pair is used in the hologram feedback equation (Equation 59), i.e., 1 means the first pair, 2 the second pair, etc.)

HOW MANY PRINTOUTS? integer

The program will perform the specified number of reconstructions (Equation 60). The program can handle up to nine image reconstructions, provided nine reference and scene pairs were input in the first section of the program.

The following is repeated for the specified number of printouts:

WHICH REFERENCE FOR RECONSTRUCTION? integer

A 1 means first reference, a 2 means second reference, etc.

IN FFT2D
OUT FFT2D

indicates inverse Fourier transform has been performed.

ONE
IN GSPLT
TWO
THREE
FOUR
FIVE
SIX
SEVEN
EIGHT

After the specified number of printouts the program will repeat the hologram iteration routine. After the specified number of iterations and program runs the program stops.

```
C FBDRVT.FR      12/21/78
C
C        THIS ROUTINE IS A DRIVER TO CREATE A HOLOGRAM
C        FEEDBACK LOOP
C
         INTEGER SNAME(7),RNAME(7),FRNAME1(7),HNAME(7),BNAME(7),ANAME(7)
         INTEGER FSNAME(7),WCONS1(7),WCONS2(7),WNAME1(7),WNAME2(7)
         INTEGER FRNAME2(7),FRNAME3(7),FRNAME4(7),FRNAME5(7),FRNAME6(7)
         INTEGER WCONS3(7),WCONS4(7),WCONS5(7),WCONS6(7)
         INTEGER WNAME3(7),WNAME4(7),WNAME5(7),WNAME6(7)
         INTEGER FRNAME7(7),FRNAME8(7),FRNAME9(7)
         COMPLEX IS(64,64),CZERO,CONE
C
         CONE=CMPLX(1.,0.)
         M=6
         NC=64
         NR=64
         IMN=1
         IAT=1
         CZERO=CMPLX(0.,0.)
         ACCEPT "NUMBER OF PROGRAM RUNS = ",NPRUNS
         DO 303 I303=1,NPRUNS
         ACCEPT "NUMBER OF POINTS TO BE AVERAGED IN GSPLT = ",NP
         IFLAG=NP+3
C
C
         ACCEPT "CONVERGENCE CONSTANT = ",C
         WRITE(10,900)
  900    FORMAT(" NAME OF HOLOGRAM FILE TO BE CREATED = ",Z)
         READ(11,905)HNAME(1)
  905    FORMAT(S7)
         DO 10 I10=1,64
         DO 10 J10=1,64
   10    IS(I10,J10)=CZERO
         CALL WBLC(IS,HNAME)
C
C        READ REFERENCE IMAGE AND STORE TRANSFORM ON DISK
C
         ACCEPT "NUMBER OF REFERENCES AND SCENE PAIRS = ",NREFS
         DO 30 I30=1,NREFS
         TYPE "REFERENCE INPUT"
         CALL SDEF(IS,M,IMN,IAT,RNAME(1))
         CALL FFT2D(IS,NC,NR,M,1)
         CALL FNORM(IS,XNORM)
         XNORM=1./XNORM
         CALL CONS(IS,XNORM)
         WRITE(10,999)
  999    FORMAT(" NAME OF REF FILE IN F DOMAIN = ",Z)
         IF(I30.EQ.1)READ(11,905)FRNAME1(1)
         IF(I30.EQ.2)READ(11,905)FRNAME2(1)
         IF(I30.EQ.3)READ(11,905)FRNAME3(1)
         IF(I30.EQ.4)READ(11,905)FRNAME4(1)
         IF(I30.EQ.5)READ(11,905)FRNAME5(1)
         IF(I30.EQ.6)READ(11,905)FRNAME6(1)
         IF(I30.EQ.7)READ(11,905)FRNAME7(1)
         IF(I30.EQ.8)READ(11,905)FRNAME8(1)
         IF(I30.EQ.9)READ(11,905)FRNAME9(1)
         IF(I30.EQ.1)CALL WBLC(IS,FRNAME1)
         IF(I30.EQ.2)CALL WBLC(IS,FRNAME2)
         IF(I30.EQ.3)CALL WBLC(IS,FRNAME3)
         IF(I30.EQ.4)CALL WBLC(IS,FRNAME4)
         IF(I30.EQ.5)CALL WBLC(IS,FRNAME5)
         IF(I30.EQ.6)CALL WBLC(IS,FRNAME6)
         IF(I30.EQ.7)CALL WBLC(IS,FRNAME7)
         IF(I30.EQ.8)CALL WBLC(IS,FRNAME8)
         IF(I30.EQ.9)CALL WBLC(IS,FRNAME9)
C
C        READ IMAGE AND STORE TRANSFORM ON DISK
C
```

```
      TYPE "SCENE INPUT"
      CALL SDEF(IS,M,IMN,IAT,SNAME(1))
      CALL FFT2D(IS,NC,NR,M,1)
      WRITE(10,998)
  998 FORMAT(" NAME OF IMAGE FILE IN F DOMAIN = ",Z)
      READ(11,905)FSNAME(1)
      CALL WBLC(IS,FSNAME)
C
C     READ NAME OF HOLOGRAM FILE TO BE CREATED
C
C
C     INPUT NAMES OF WORKING DISK FILES
C
      WRITE(10,1000)
 1000 FORMAT( " NAME OF WORKING FILE = ",Z)
      IF(I30 .EQ. 1)READ(11,905)WNAME1(1)
      IF(I30 .EQ. 2)READ(11,905)WNAME2(1)
      IF(I30 .EQ. 3)READ(11,905)WNAME3(1)
      IF(I30 .EQ. 4)READ(11,905)WNAME4(1)
      IF(I30 .EQ. 5)READ(11,905)WNAME5(1)
      IF(I30 .EQ. 6)READ(11,905)WNAME6(1)
      WRITE(10,1001)
 1001 FORMAT(" NAME OF CONSTANT FILE = ",Z)
      IF(I30 .EQ. 1)READ(11,905)WCONS1(1)
      IF(I30 .EQ. 2)READ(11,905)WCONS2(1)
      IF(I30 .EQ. 3)READ(11,905)WCONS3(1)
      IF(I30 .EQ. 4)READ(11,905)WCONS4(1)
      IF(I30 .EQ. 5)READ(11,905)WCONS5(1)
      IF(I30 .EQ. 6)READ(11,905)WCONS6(1)
C
C     PERFORM R*R CONJUGATE
C
      IF(I30 .EQ. 1)CALL RBLC(IS,FRNAME1)
      IF(I30 .EQ. 2)CALL RBLC(IS,FRNAME2)
      IF(I30 .EQ. 3)CALL RBLC(IS,FRNAME3)
      IF(I30 .EQ. 4)CALL RBLC(IS,FRNAME4)
      IF(I30 .EQ. 5)CALL RBLC(IS,FRNAME5)
      IF(I30 .EQ. 6)CALL RBLC(IS,FRNAME6)
      IF(I30 .EQ. 1)CALL HCRE(IS,FRNAME1)
      IF(I30 .EQ. 2)CALL HCRE(IS,FRNAME2)
      IF(I30 .EQ. 3)CALL HCRE(IS,FRNAME3)
      IF(I30 .EQ. 4)CALL HCRE(IS,FRNAME4)
      IF(I30 .EQ. 5)CALL HCRE(IS,FRNAME5)
      IF(I30 .EQ. 6)CALL HCRE(IS,FRNAME6)
      CALL CONS(IS,C)
C
C     FIND 1/(1 + C * R*R CONJ)
C
      DO 20 I20=1,NC
      DO 20 J20=1,NR
   20 IS(I20,J20)=CONE/(CONE+IS(I20,J20))
C
      IF(I30 .EQ. 1)CALL WBLC(IS,WCONS1)
      IF(I30 .EQ. 2)CALL WBLC(IS,WCONS2)
      IF(I30 .EQ. 3)CALL WBLC(IS,WCONS3)
      IF(I30 .EQ. 4)CALL WBLC(IS,WCONS4)
      IF(I30 .EQ. 5)CALL WBLC(IS,WCONS5)
      IF(I30 .EQ. 6)CALL WBLC(IS,WCONS6)
C
C     FIND C*RCONJ*S/(    )
C
      CALL RBLC(IS,FSNAME)
      IF(I30 .EQ. 1)CALL HCRE(IS,FRNAME1)
      IF(I30 .EQ. 2)CALL HCRE(IS,FRNAME2)
      IF(I30 .EQ. 3)CALL HCRE(IS,FRNAME3)
      IF(I30 .EQ. 4)CALL HCRE(IS,FRNAME4)
      IF(I30 .EQ. 5)CALL HCRE(IS,FRNAME5)
      IF(I30 .EQ. 6)CALL HCRE(IS,FRNAME6)
      CALL CONS(IS,C)
C
      IF(I30 .EQ. 1)CALL HRCON(IS,WCONS1)
      IF(I30 .EQ. 2)CALL HRCON(IS,WCONS2)
      IF(I30 .EQ. 3)CALL HRCON(IS,WCONS3)
      IF(I30 .EQ. 4)CALL HRCON(IS,WCONS4)
      IF(I30 .EQ. 5)CALL HRCON(IS,WCONS5)
```

```
      IF(I30 .EQ. 6)CALL HRCON(IS,WCONS6)
      IF(I30 .EQ. 1)CALL WBLC(IS,WNAME1)
      IF(I30 .EQ. 2)CALL WBLC(IS,WNAME2)
      IF(I30 .EQ. 3)CALL WBLC(IS,WNAME3)
      IF(I30 .EQ. 4)CALL WBLC(IS,WNAME4)
      IF(I30 .EQ. 5)CALL WBLC(IS,WNAME5)
      IF(I30 .EQ. 6)CALL WBLC(IS,WNAME6)
C
   30 CONTINUE
      ACCEPT " NUMBER OF ITERATIONS = ",NRUNS
      ACCEPT "NUMBER OF ITERATIONS PER PRINTOUT = ",IPF
      IPFLAG=0
      DO 200 I200=1,NRUNS
      ACCEPT "WHICH REFERENCE AND SCENE = ",NX
C
C     READ H(N-1)
C
      CALL RBLC(IS,HNAME)
C
C     MULTIPLY BY (1)/(1 + C*ABS(R)**2)
C
      IF(NX .EQ. 1)CALL HRCON(IS,WCONS1)
      IF(NX .EQ. 2)CALL HRCON(IS,WCONS2)
      IF(NX .EQ. 3)CALL HRCON(IS,WCONS3)
      IF(NX .EQ. 4)CALL HRCON(IS,WCONS4)
      IF(NX .EQ. 5)CALL HRCON(IS,WCONS5)
      IF(NX .EQ. 6)CALL HRCON(IS,WCONS6)
C
C     ADD WEIGHTED SCENE HOLOGRAM
C
      IF(NX .EQ. 1)CALL HADD(IS,WNAME1)
      IF(NX .EQ. 2)CALL HADD(IS,WNAME2)
      IF(NX .EQ. 3)CALL HADD(IS,WNAME3)
      IF(NX .EQ. 4)CALL HADD(IS,WNAME4)
      IF(NX .EQ. 5)CALL HADD(IS,WNAME5)
      IF(NX .EQ. 6)CALL HADD(IS,WNAME6)
C
C     WRITE H(N) TO DISK
C
      CALL WBLC(IS,HNAME)
      IF(IPFLAG .NE. 0)GO TO 201
      IPFLAG=IPF
C
C     PERFORM HOLOGRAM RECONSTRUCTION
C
      ACCEPT "HOW MANY PRINTOUTS ? ",NPRIN
      DO 330 I330 =1,NPRIN
      ACCEPT "WHICH REFERENCE FOR RECONSTRUCTION ? ",NXX
      CALL RBLC(IS,HNAME)
      IF(NXX .EQ. 1)CALL HRCON(IS,FRNAME1)
      IF(NXX .EQ. 2)CALL HRCON(IS,FRNAME2)
      IF(NXX .EQ. 3)CALL HRCON(IS,FRNAME3)
      IF(NXX .EQ. 4)CALL HRCON(IS,FRNAME4)
      IF(NXX .EQ. 5)CALL HRCON(IS,FRNAME5)
      IF(NXX .EQ. 6)CALL HRCON(IS,FRNAME6)
      IF(NXX .EQ. 7)CALL HRCON(IS,FRNAME7)
      IF(NXX .EQ. 8)CALL HRCON(IS,FRNAME8)
      IF(NXX .EQ. 9)CALL HRCON(IS,FRNAME9)
C
      CALL FFT2D(IS,NC,NR,M,2)
      CALL CSPLT(IS,NC,NR,IFLAG)
  330 CONTINUE
C
C
C     LOOP TO DIFFERENCE FOR NEXT ITERATION
C
  201 IPFLAG=IPFLAG-1
  200 CONTINUE
  303 CONTINUE
      CALL RESET
      STOP
      END
```

```
C CRSUM.FR       11/3/78
C
C        THIS PROGRAMS WRITES A BLOCK OF COMPLEX ZEROS ON
C        A DISK FILE
CC
C                SUBROUTINES USED:  WBLKC.RB
C
         COMPLEX IS(64,64),CZERO
         CZERO=CMPLX(0.,0.)
         DO 10 I=1,64
         DO 10 J=1,64
         IS(I,J)=CZERO
   10    CONTINUE
         CALL WBLKC(IS,NAME)
         STOP
         END
```

4.2.7 RBLC.FR

RBLC.FR is a FORTRAN subroutine that reads a 64 by 64 complex array from disk as illustrated in Figure 20. The routine uses channel 1 for the disk transfer.

Two arguments are passed in the call statement. The first is the name of the complex array in memory which will receive the disk data. The second is an integer array that contains the file name of the data on disk. The CALL statement is of the form

CALL RBLC (arrayname, filename array)

The subroutine assumes the data on disk is a 64 by 64 complex array with the first row of data in BLOCK 1 of the file.

4.2.8 RDBLC.FR

RDBLC.FR is a FORTRAN subroutine that reads a file name from the console, and then reads a 64 by 64 complex array from the specified file (see Figure 21).

The subroutine outputs to the console the message:

"NAME OF FILE TO BE READ = "

to which the operator responds with a file name (seven characters maximum). A 64 by 64 complex array is then read from the specified disk file. The subroutine assumes the data on disk is a 64 by 64 complex array with the first row of data in BLOCK 1 of the file.

The complex array is returned to the calling program. The CALL statement is of the form

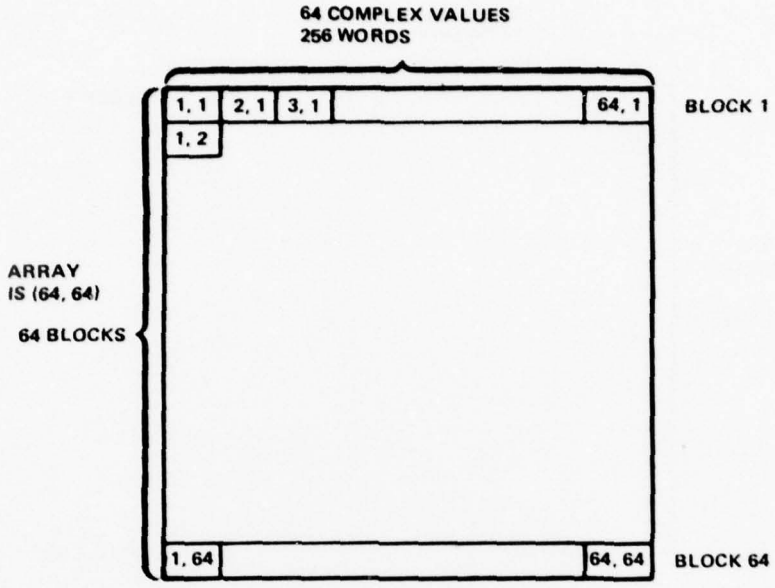
CALL RBLKC (arrayname)

4.2.9 WBLC.FR

WBLC.FR is the name of a FORTRAN subroutine that is used to write a 64 by 64 complex array out to disk (see Figure 22). The subroutine uses channel 1 for the disk transfer.

Two arguments are passed to the subroutine. The first is the array name of the complex data which is to be written out to disk. The second is an integer array which contains the file name of the disk file to be created. The CALL statement is of the form:

CALL WBLC (arrayname, filename array)



- KNAME ( ) IS READ FROM CONSOLE
- FILE KNAME ( ) IS READ FROM DISK INTO ARRAY IS (,)
- IS (,) IS PASSED TO CALLING PROGRAM

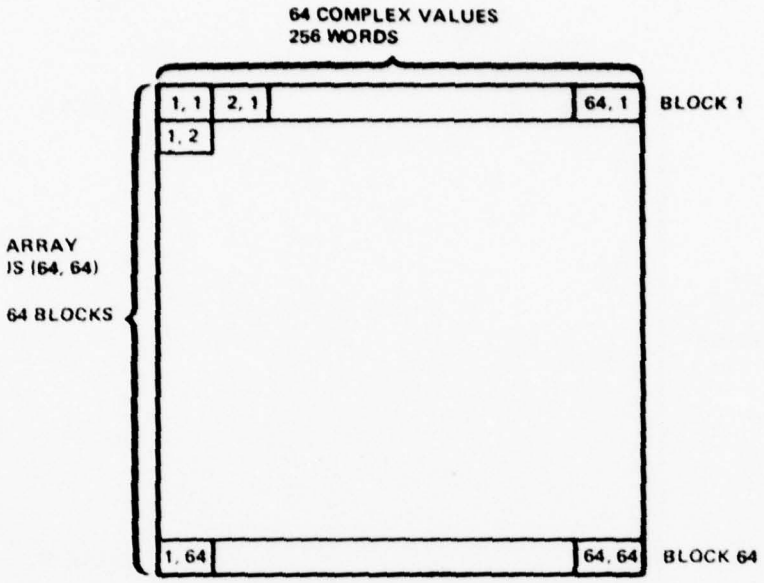
```
C   RBLKC.FR          10/23/78
C
        SUBROUTINE RBLKC(IS)
C
C       THIS SUBROUTINE READS A 64 X 64 COMPLEX ARRAY
C
        COMPLEX IS(64,64)
        DIMENSION KNAME(7)

C

        WRITE(10,900)
900     FORMAT(" NAME OF FILE TO BE READ = ",Z)
        READ(11,905) KNAME(1)
905     FORMAT(S7)
        CALL OPEN(1,KNAME(1),2,IER,512)
        IF(IER.NE.1) STOP --OPEN ERROR  --
        NBLOCK=64
        IR=1
        CALL RDBLK(1,IR,IS,NBLOCK,IER)
        IF(IER .NE. 1)STOP --RRBLK ERR--
        CALL CLOSE(1,IER)
        IF(IER .NE. 1)STOP --CLOSE ERR--
        RETURN
        END
```

9385

FIGURE 20. RBLKC.FR



- KNAME ( ) IS PASSED TO SUBROUTINE
- IS (,) IS READ FROM DISK FILE KNAME ( ) AND PASSED TO CALLING PROGRAM
- KNAME IS RETURNED TO CALLING PROGRAM UNCHANGED

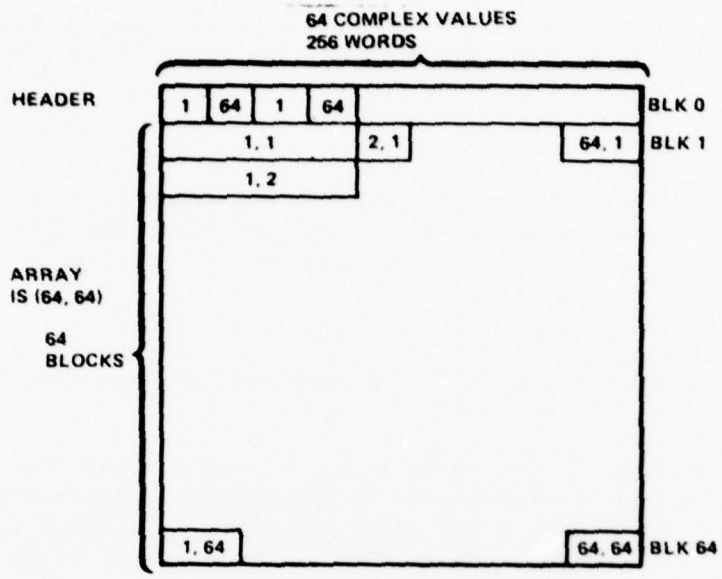
```
C    RBLC.FR          12/6/78
C
          SUBROUTINE RBLC(IS,KNAME)
C
C         THIS SUBROUTINE READS A 64 X 64 COMPLEX ARRAY
C
          COMPLEX IS(64,64)
          DIMENSION KNAME(7)

C

          CALL OPEN(1,KNAME(1),2,IER,512)
          IF(IER .NE. 1) STOP --OPEN ERROR  --
          NBLOCK=64
          IR=1
          CALL RDBLK(1,IR,IS,NBLOCK,IER)
          IF(IER  .NE.  1)STOP --RRBLK ERR--
          CALL CLOSE(1,IER)
          IF(IER  .NE.  1)STOP --CLOSE ERR--
          RETURN
          END
```

9386

FIGURE 21. RBLC.FR



- IS (64, 64) PASSED TO SUBROUTINE
- KNAME (7) PASSED TO CALLING PROGRAM
  IS (64, 64) RETURNED TO CALLING
  PROGRAM UNCHANGED

```
C   WBLKC.FR          10/23/78
C
      SUBROUTINE WBLKC(IS,KNAME)
C
C     THIS SUBROUTINE WRITES A 64 X 64 COMPLEX ARRAY
C
      COMPLEX IS(64,64)
      DIMENSION IDD(256),KNAME(7)

C

      WRITE(10,900)
900   FORMAT(" NAME OF FILE TO BE CREATED = ",Z)
      READ(11,905) KNAME(1)
905   FORMAT(S7)
      CALL DFILW(KNAME(1),IER)
      CALL CFILW(KNAME(1),2,IER)
      IF(IER.NE.1) STOP -- CFILW ERROR (BGD) --
      CALL OPEN(1,KNAME(1),2,IER,512)
      IF(IER.NE.1) STOP --OPEN ERROR (BGD) --
      IDD(2)=64
      IDD(3)=1
      IDD(4)=64
      IDD(5)=1
C
C     WRITE HEADER INFORMATION
C
      IR=0
      CALL WRBLK(1,IR,IDD,1,IER)
      IF(IER .NE. 1)STOP --WRBLK ERR--
      NBLOCK=64
      IR=1
      CALL WRBLK(1,IR,IS,NBLOCK,IER)
      IF(IER .NE. 1)STOP --WRBLK ERR--
      CALL CLOSE(1,IER)
      IF(IER .NE. 1)STOP --CLOSE ERR-
      RETURN
      END
```

9387

FIGURE 22. WBLKC.FR

The subroutine writes some header information (number of rows and columns) into BLOCK Ø of the file, then writes the array out to disk with the first row of the array in BLOCK 1 of the file. Subsequent rows are written into subsequent blocks.

4.2.10 WBLKC.FR

WBLKC.FR is a FORTRAN subroutine that accepts a file name from the console and writes a 64 by 64 complex array out to the specified disk file (see Figure 23). The subroutine outputs to the console the message:

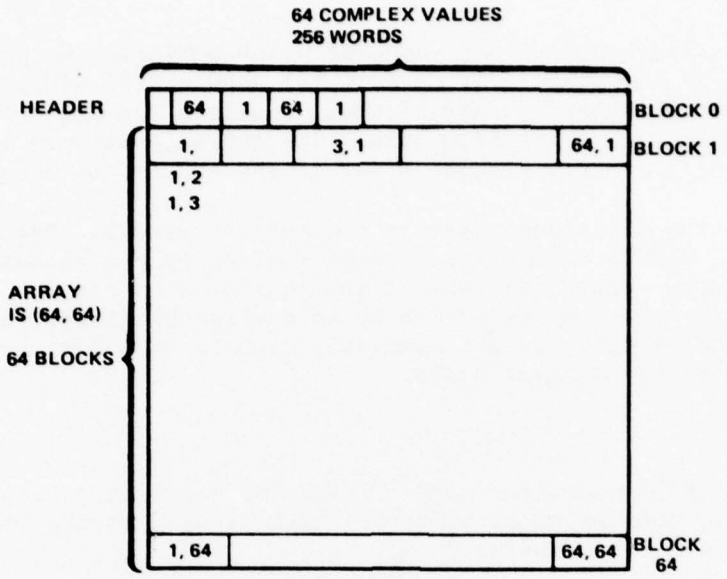
"NAME OF FILE TO BE CREATED = "

to which the operator responds with a file name and carriage return (seven characters maximum)

The subroutine writes some header information (number of rows and columns) into BLOCK Ø of the array, then writes the 64 by 64 complex array (passed from the calling program) out to disk with the first row of data in BLOCK 1 of the specified file. Subsequent rows are written into subsequent blocks.

The subroutine returns an integer array to the calling program containing the specified disk file name. The CALL statement is of the form:

CALL WBLKC (arrayname, filename array)



- AUGMENTS IS AND KNAME PASSED TO SUBROUTINE
- IS AND KNAME RETURNED TO CALLING PROGRAM UNCHANGED

```
C WBLC.FR          12/6/78
C
      SUBROUTINE WBLC(IS,KNAME)
C
C     THIS SUBROUTINE WRITES A 64 X 64 COMPLEX ARRAY
C
      COMPLEX IS(64,64)
      DIMENSION IDD(256),KNAME(7)

C

      CALL DFILW(KNAME(1),IER)
      CALL CFILW(KNAME(1),2,IER)
      IF(IER.NE.1) STOP -- CFILW ERROR (BQD) --
      CALL OPEN(1,KNAME(1),2,IER,512)
      IF(IER.NE.1) STOP --OPEN ERROR (BQD) --
      IDD(2)=64
      IDD(3)=1
      IDD(4)=64
      IDD(5)=1
C
C     WRITE HEADER INFORMATION
C
      IR=0
      CALL WRBLK(1,IR,IDD,1,IER)
      IF(IER .NE. 1)STOP --WRBLK ERR--
      NBLOCK=64
      IR=1
      CALL WRBLK(1,IR,IS,NBLOCK,IER)
      IF(IER .NE. 1)STOP --WRBLK ERR--
      CALL CLOSE(1,IER)
      IF(IER .NE. 1)STOP --CLOSE ERR-
      RETURN
      END
```

9388

FIGURE 23. WBLC.FR

4.3 UTILITY SUBROUTINES

There are six utility subroutines, each of which performs one simple operation on a complex 64 by 64 array: (1) DIFF.—, (2) HADD.—, (3) HCRE.—, (4) HRCON.—, (5) FNORM.—, and (6) CONS.—. Each of these routines use channel 1 if data needs to be transferred to or from disk. The subroutines open and close channel 1, and will cause a program ABORT if any I/O errors occur on channel 1.

DIFF.— computes the difference between two complex arrays. HADD.— adds two complex arrays. CONS.— multiplies a complex array by a real number. FNORM.— computes the square root of the sum of the squares of a complex array. HRCON.— multiplies two complex arrays on an element by element basis. HCRE.— multiplies a complex array by the complex conjugate of a different complex array on an element by element basis.

4.3.1 DIFF.FR

DIFF.FR is a FORTRAN subroutine named DIFF. The subroutine takes the difference of two complex 64 by 64 arrays element by element, one array being in memory and the other array on disk.

Two parameters are passed to the subroutine. The first is the 64 by 64 complex array (in memory). The second is an integer array containing the file name of the complex array residing on disk. The call statement is of the form

```
CALL DIFF (arrayname, filename array)
```

The subroutine assumes the disk data is a 64 by 64 complex array with the 1st row of data in BLOCK 1 of the disk file.

The data in memory is subtracted from the data on disk, with the result stored in the 64 by 64 complex array and returned to the calling program.

4.3.2 HADD.FR

HADD.FR is a FORTRAN subroutine named HADD. The subroutine adds, element by element, a 64 by 64 complex array in memory, with a 64 by 64 complex array on disk. Channel 1 is used by the subroutine.

Two parameters are passed to the subroutine. The first is the 64 by 64 complex array in memory. The second is an integer array containing the file name of the complex array residing on disk. The call statement is of the form:

```
CALL HADD (arrayname, filename array)
```

The subroutine assumes the disk data is a 64 by 64 complex array with the first row of data in BLOCK 1 of the disk file. Subsequent rows are in subsequent blocks.

The two arrays are added, with the resulting 64 by 64 complex array returned to the calling program.

4.3.3 FNORM.FR

FNORM.FR is a FORTRAN subroutine named FNORM. The subroutine calculates the square root of the sum of the squares of the elements in a 64 by 64 complex array. The result can be used to normalize the complex array.

Two arguments are passed in the CALL statement. The first is the 64 by 64 complex array. The second argument is the square root of the sum of the squares returned by the subroutine to the calling program. The CALL statement is of the form:

```
CALL FNORM (arrayname, realvariable)
```

The 64 by 64 complex array is returned to the calling program unchanged.

4.3.4 CONS.FR

CONS.FR is a FORTRAN subroutine named CONS. The subroutine multiplies each element of a 64 by 64 complex array by a real variable.

Two arguments are passed in the CALL statement. The first is the 64 by 64 complex array. The second is the real variable. The CALL statement is of the form

```
CALL CONS (arrayname, realvariable)
```

The resulting 64 by 64 complex array is returned to the calling program.

4.3.5 HRCON.FR

HRCON.FR is a FORTRAN subroutine named HRCON. The subroutine multiplies a 64 by 64 complex array in memory with one on disk, on an element by element basis.

Two arguments are passed from the calling program to the subroutine. The first is the 64 by 64 complex array in memory. The second is an integer array containing the file name of the disk file. The CALL statement is of the form

```
CALL HRCON (arrayname, filename array)
```

The subroutine assumes the disk file is a 64 by 64 complex array with the first row of data in BLOCK 1 of the disk file, with sequential rows in sequential file blocks.

The resulting complex array is returned to the calling program.

4.3.6 HCRE.FR

HCRE.FR is a FORTRAN subroutine named NCRE. The subroutine multiplies a 64 by 64 complex array in memory with the complex conjugate of a 64 by 64 complex array on disk, on an element by element basis.

Two arguments are passed from the calling program to the subroutine. The first is the 64 by 64 complex array in memory. The second is an integer array containing the file name of the disk file. The CALL statement is of the form:

```
CALL HCRE (arrayname, filename array)
```

The subroutine assumes the disk file is a 64 by 64 complex array with the first row of data in BLOCK 1 of the file.

The subroutine returns the resulting 64 by 64 complex array to the calling program.

```
C DIFF.FR        12/5/78
C
C       THIS SUBROUTINE TAKES THE DIFFERNCE OF TWO ARRAYS
C       ONE PASSED IN THE ARGUMENT LIST AND THE OTHER ON
C       DISK
C
        SUBROUTINE DIFF(IS,CSNAME)
        COMPLEX IS(64,64),CTEMP(64)
        INTEGER CSNAME(7)
C
        CALL OPEN(1,CSNAME,2,IER,512)
        IF(IER .NE. 1)STOP --S(X) OPEN ERROR--
C
        NBLOCK=1
C
        DO 10 I10=1,64
        IR=I10
        CALL RDBLK(1,IR,CTEMP,NBLOCK,IER)
        IF(IER .NE. 1)STOP --RDBLK ERROR--
        DO 20 J20=1,64
        IS(J20,I10)=CTEMP(J20)-IS(J20,I10)
   20   CONTINUE
   10   CONTINUE
        CALL CLOSE(1,IER)
        IF(IER .NE. 1)STOP --CLOSE ERROR--
        RETURN
        END
```

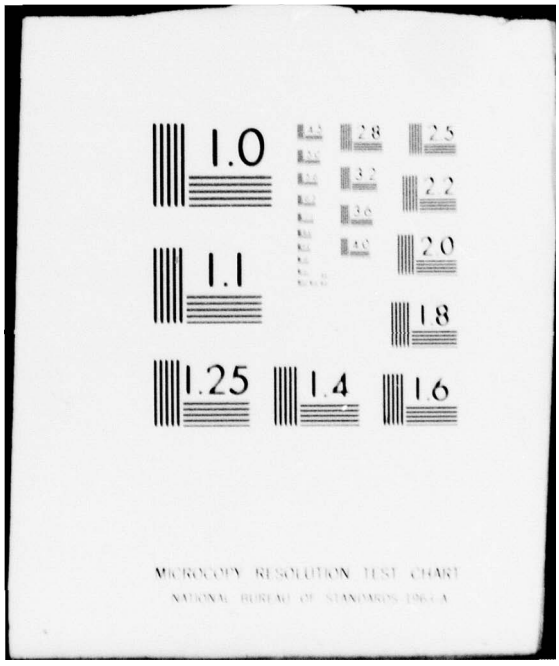
```
C HADD.FR        12/5/78
C
C        THIS SUBROUTINE ADDS TWO ARRAYS, ONE IN MEMORY
C        WITH ONE ON DISK
C
         SUBROUTINE HADD(IS,CSNAME)
         COMPLEX IS(64,64),CTEMP(64)
         INTEGER CSNAME(7)
C
         CALL OPEN(1,CSNAME,2,IER,512)
         IF(IER .NE. 1)STOP --S(X) OPEN ERROR--
C
         NBLOCK=1
C
         DO 10 I10=1,64
         IR=I10
         CALL RDBLK(1,IR,CTEMP,NBLOCK,IER)
         IF(IER .NE. 1)STOP --RDBLK ERROR--
         DO 20 J20=1,64
         IS(J20,I10)=IS(J20,I10)+CTEMP(J20)
   20    CONTINUE
   10    CONTINUE
         CALL CLOSE(1,IER)
         IF(IER .NE. 1)STOP --CLOSE ERROR--
         RETURN
         END
```

```
C FNORM.FR       12/7/78
C
C        THIS SUBROUTINE FINDS THE SQRT OF THE
C        SUM OF THE SQUARES OF ELEMENTS IN
C        A COMPLEX ARRAY
C
         SUBROUTINE FNORM(IS,XNORM)
         COMPLEX IS(64,64),CSUM
C
         CSUM=CMPLX(0.,0.)
         DO 10 I10=1,64
         DO 10 J10=1,64
         CSUM=CSUM+IS(I10,J10)*IS(I10,J10)
   10    CONTINUE
         XNORM=SQRT(CABS(CSUM))
         RETURN
         END
```

AD-A068 638 FORD AEROSPACE AND COMMUNICATIONS CORP NEWPORT BEACH --ETC F/G 14/5
MACHINE HOLOGRAPHY.(U)
MAR 79 C L RICHARDS N00014-78-C-0659
UNCLASSIFIED U-6516 NL

2 OF 2
AD
A068638

END
DATE
FILMED
6-79
DDC


1.0
1.1
1.25
1.4
1.6
2.8
3.2
3.6
2.5
2.2
2.0
1.8
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

```
C CONS.FR        12/5/78
C
C         THIS SUBROUTINE MULTIPLIES A COMPLEX ARRAY BY A CONSTANT
C
          SUBROUTINE CONS(IS,C)
          COMPLEX IS(64,64),COMC
C
          COMC=CMPLX(C,0.)
C
          DO 10 I10=1,64
          DO 20 J20=1,64
          IS(J20,I10)=IS(J20,I10)*COMC
   20     CONTINUE
   10     CONTINUE
          RETURN
          END
```

```
C HRCON.FR       12/5/78
C
C        THIS SUBROUTINE MULTIPLIES AN ARRAY IN MEMORY
C        BY ONE ON DISK,,ELEMENT BY ELEMENT.
         SUBROUTINE HRCON(IS,CSNAME)
         COMPLEX IS(64,64),CTEMP(64)
         INTEGER CSNAME(7)
C
         CALL OPEN(1,CSNAME,2,IER,512)
         IF(IER .NE. 1)STOP --S(X) OPEN ERROR--
C
         NBLOCK=1
C
         DO 10 I10=1,64
         IR=I10
         CALL RDBLK(1,IR,CTEMP,NBLOCK,IER)
         IF(IER .NE. 1)STOP --RDBLK ERROR--
         DO 20 J20=1,64
         IS(J20,I10)=IS(J20,I10)*CTEMP(J20)
   20    CONTINUE
   10    CONTINUE
         CALL CLOSE(1,IER)
         IF(IER .NE. 1)STOP --CLOSE ERROR--
         RETURN
         END
```

```
C HCRE.FR        12/5/78
C
C       THIS SUBROUTINE MULTIPLIES AN ARRAY IN MEMORY
C       BY THE CONJUGATE OF AN ARRAY ON DISK TO
C       PRODUCE A HOLOGRAM
C
        SUBROUTINE HCRE(IS,CSNAME)
        COMPLEX IS(64,64),CTEMP(64)
        INTEGER CSNAME(7)
C
        CALL OPEN(1,CSNAME,2,IER,512)
        IF(IER .NE. 1)STOP --S(X) OPEN ERROR--
C
        NBLOCK=1
C
        DO 10 I10=1,64
        IR=I10
        CALL RDBLK(1,IR,CTEMP,NBLOCK,IER)
        IF(IER .NE. 1)STOP --RDBLK ERROR--
        DO 20 J20=1,64
        IS(J20,I10)=IS(J20,I10)*CONJG(CTEMP(J20))
   20   CONTINUE
   10   CONTINUE
        CALL CLOSE(1,IER)
        IF(IER .NE. 1)STOP --CLOSE ERROR--
        RETURN
        END
```

```
CFFT2D.FR       04/11/78
        SUBROUTINE FFT2D(A,NC,NR,M,INV)
        COMPLEX A(64,64),W(6,2),T,U
        TYPE"IN FFT2D"
        N=2**M
        FN=(1./N)**2
        NV2=N/2
        NM1=N-1
        DO 50 L=1,M
        ANG=-2.*3.141596/2**L
        W(L,1)=CMPLX(COS(ANG),SIN(ANG))
50      W(L,2)=CMPLX(COS(-ANG),SIN(-ANG))
        DO 26 J2=1,2
        J=1
        DO 7 I=1,NM1
        IF(I.GE.J) GO TO 5
        DO 4 KK=1,N
        T=A(J,KK)
        A(J,KK)=A(I,KK)
4       A(I,KK)=T
5       K=NV2
6       IF(K.GE.J) GO TO 7
        J=J-K
        K=K/2
        GO TO 6
7       J=J+K
        LE=1
        DO 20 L=1,M
        LE1=LE
        LE=LE+LE
        U=(1.,0.)
        DO 20 J=1,LE1
        DO 10 I=J,N,LE
        IP=I+LE1
        DO 10 KK=1,N
        T=A(IP,KK)*U
        A(IP,KK)=A(I,KK)-T
10      A(I,KK)=A(I,KK)+T
20      U=U*W(L,INV)
        DO 25 I=1,N
        DO 25 KK=I,N
        T=A(I,KK)
        A(I,KK)=A(KK,I)
25      A(KK,I)=T
26      CONTINUE
        IF(INV.EQ.1) GO TO 55
        IF(IFT .EQ. 2) FN=FN/(N**2)
        DO 30 I=1,N
        DO 30 KK=1,N
30      A(I,KK)=A(I,KK)*FN
55      TYPE"OUT FFT2D"
        RETURN
        END
```

```
C    GSPLT.FR      5/10/78          REV. A  11/1/78
C                                   REV. B 11/2/78
C                                   REV. C  11/28/78
C
      SUBROUTINE GSPLT(IS,NC,NR,GFLG)
C
C     GSPLT MAKES A GREY SCALE PLOT OF THE MAGNITUDE
C     OF THE NR X NC COMPLEX ARRAY IS
C     GFLG= 0  NEW LINEAR GRAY SCALE
C           1  PREVIOUS GRAY SCALE
C           2  LOG GRAY SCALE
C           3  NORMALIZE PEAK VALUE TO 99
C           4  NORMALIZE PEAK VALUE TO 1000
      COMMON/GG/VAL(10),GRAT(30)
      COMMON/JJ/LBUF(132,3)
      DIMENSION KBUF(260),TAB(1024),AVE(64)
      INTEGER TAB,VAL,GRAT,GFLG
      COMPLEX IS(64,64),DD(64),CPKMG,CPNORM,CSUM
C
      DATA LBUF(131,1)/5,12K/
      DATA LBUF(131,2)/5,12K/
      DATA LBUF(131,3)/5,12K/
      TYPE "ONE"
C
C
C     INPUT NUMBER OF POINTS TO BE AVERAGED
C     ZERO AVE ARRAY
C
      ACCEPT "NUMBER OF POINTS AVERAGED",NAVE
      DO 202 I202=1,64
      AVE(I202)=0.
  202 CONTINUE
C
C
      TYPE"IN GSPLT"
C
      DATA GRAT/100K,100K,122K,111K,122K,155K,4*177K,
     *  100K,122K,100K,144K,155K,3*122K,155K,177K,
     *  100K,100K,122K,111K,122K,155K,4*177K/
C
      CALL OPEN(2,"SCRATCH",0,IER,512)
C
      DO 1 I1=1,260
1     KBUF(I1)=0
C
      DO 23 I1=1,NR
      DO 21 I2=1,NC
21    DD(I2)=IS(I2,I1)
      CALL WRBLK(2,I1-1,DD,1,IER)
      IF(IER.NE.1) STOP --WRBLK ERR--
23    CONTINUE
30    CONTINUE
      TYPE "TWO"
      PKMG=0.
      DO 5 J=1,NR
      DO 5 I=1,NC
      SS=CABS(IS(I,J))
      IF(GFLG .EQ. 2) SS=10.*ALOG10(SS+1.)
      IF(SS.GT.PKMG) PKMG=SS
5     CONTINUE
C
      TYPE "THREE"
C     NORMALIZE  INPUT ARRAY
C
      IF(GFLG .LT. 3)GO TO 87
      PNORM=1000.
      IF(GFLG .EQ. 3)PNORM=99.
      CPNORM=CMPLX(PNORM,0.)
      CPKMG=CMPLX(PNORM/PKMG,0.)
C
```

```
      DO 88 I88=1,NC
      DO 88 J88=1,NR
      IS(I88,J88)=IS(I88,J88)*CPKMG
C
C     FIND THE "NAVE" NUMBER OF PEAK POINTS
C
      SS=CABS(IS(I88,J88))
      IF(SS .LE. AVE(NAVE))GO TO 204
      AVE(NAVE)=SS
      NAVEL=NAVE-1
      DO 206 I206=1,NAVEL
      IF(AVE(NAVE+1-I206) .LE. AVE(NAVE-I206))GO TO 204
      TEMP=AVE(NAVE-I206)
      AVE(NAVE-I206)=AVE(NAVE+1-I206)
      AVE(NAVE+1-I206)=TEMP
  206 CONTINUE
  204 CONTINUE
C
C
   88 CONTINUE
      TYPE "FOUR"
C
C
C     CALCULATE STATISTICS OF BACKGROUND
C
   87 CONTINUE
C
      CSUM=CMPLX(0.,0.)
      BSUM=0
      BSUM2=0
C
      DO 89 I89=1,NC
      A=CABS(IS(I89,1))
      B=CABS(IS(I89,NR))
      CSUM=CSUM+IS(I89,1)+IS(I89,NR)
      IS(I89,1)=CPNORM
      IS(I89,NR)=CPNORM
      BSUM2=BSUM2+A*A+B*B
   89 CONTINUE
      TYPE "FIVE"
      NROWS=NR-1
C
      DO 91 I91=2,NROWS
      A=CABS(IS(1,I91))
      B=CABS(IS(NR,I91))
      CSUM=CSUM+IS(1,I91)+IS(NC,I91)
      IS(1,I91)=CPNORM
      IS(NC,I91)=CPNORM
      BSUM2=BSUM2+A*A+B*B
   91 CONTINUE
      TYPE "SIX"
C
C
C     CALCULATE AVERAGE OF THE PEAK VALUES
C
      PAVE=0.
      DO 208 I208=1,NAVE
      PAVE=PAVE+AVE(I208)
  208 CONTINUE
      PAVE=PAVE/FLOAT(NAVE)
C
C
      BSUM=CABS(CSUM)
      BMEAN=BSUM/(2*NC+2*NR-4)
      BVARIANCE=(BSUM2/FLOAT(2*NC+2*NR-4))-BMEAN*BMEAN
      BSTDDEV=SQRT(BVARIANCE)
      BSIGNOI=PAVE/BSTDDEV
```

```
CC
C
C
C
C       PLOT MAGNITUDE
C
        DO 92 I92=1,NC
        DO 92 J92=1,NR
        SS=CABS(IS(I92,J92))
        IS(I92,J92)=CMPLX(SS,0.)
   92   CONTINUE
        IFLG=0
        SUM1=0.
        SUM2=0
        NN=0
        TYPE "SEVEN"
        DO 8 I6=1,NR
        DO 8 I5=1,NC
        SS=REAL(IS(I5,I6))
        IF(SS.LE.0.) SS=0.
        IS(I5,I6)=CMPLX(SS,0.)
C       DISCOUNT 0 BACKGROUND IN GRAYSCALE DEFINITION
        IF(SS .GT. 0.) NN=NN+1
        SUM1=SUM1+SS
8       SUM2=SUM2+SS**2
C
        TYPE "EIGHT"
        RMEAN=SUM1/NN
        RSTD=SQRT((SUM2-SUM1**2/NN)/(NN-1))
        DELTA=RSTD/2.
C       CHECK FOR MAGNITUDE TOO HIGH FOR LINEAR PLOT
        UP=RMEAN+SDEV
        IF(UP .LT. 1023) GO TO 35
        IF(GFLG .EQ. 2) GO TO 35
        TYPE"SWITCH TO LOG PLOT"
        GFLG=2
        GO TO 30
35      CONTINUE
C
C       MAKE A NEW GRAY SCALE
C
        IF(GFLG .EQ. 1) GO TO 3
        DO 10 I=1,10
        KTEMP=IFIX(RMEAN+FLOAT(I-5)*DELTA)
        IF(KTEMP.LT.1) KTEMP=1
        IF(KTEMP.GT.1023) KTEMP=1023
10      VAL(I)=KTEMP
        DO 12 I=1,10
        VAL(I)=100.*FLOAT(I)
        IF(GFLG .EQ. 3)VAL(I)=FLOAT(I-1)*10.+9.
   12   CONTINUE
C
C
        J=1
        DO 11 I=1,1024
        IF(VAL(J).LT.I) J=J+1
        IF(J.GT.10) J=10
11      TAB(I)=J
3       CONTINUE
C
        NNC=NC
        NNR=NR
        IF(GFLG .EQ. 0 .OR. GFLG .EQ. 1)WRITE(12,905)
905     FORMAT(///" LINEAR PLOT");
        IF(GFLG .EQ. 2) WRITE(12,910)
910     FORMAT(///" LOG PLOT")
        WRITE(12,915) PKMG,RMEAN,RSTD
```

```
915     FORMAT(" MAX=",F6.0," MEAN=",F6.0,"SDEV=",F6.0)
        WRITE(12,900)NNR,NNC,VAL
900     FORMAT(1X,"NROWS=",I4,
     &  " NCOLS=",I4/1X,"GRAY SCALE=",10I5//)
C
        DO 15 I1=1,NR
        DO 20 I2=1,NC
        KTEMP=REAL(IS(I2,I1))
        IF(KTEMP.GT.1023) KTEMP=1023
        IF(KTEMP.GT.VAL(10)) KTEMP=VAL(10)
20      KBUF(I2+10)=KTEMP
C
        CALL PICPLT(KBUF,260,VAL,GRAT,LBUF,TAB)
        CALL PLPT(LBUF(1,1))
        CALL PLPT(LBUF(1,2))
        CALL PLPT(LBUF(1,3))
15      CONTINUE
C
        DO 27 I1=1,NR
        CALL RDBLK(2,I1-1,DD,1,IER)
        IF(IER.NE.1) STOP --RDBLK ERR--
        DO 25 I2=1,NC
25      IS(I2,I1)=DD(I2)
27      CONTINUE
        CALL CLOSE(2,IER)
        WRITE(12,441)PKMG,BMEAN,BSTDDEV,BSIGNOI
  441   FORMAT(1HO,5HPEAK=,2X,E14.4/1X,5HMEAN=,2X,E14.4,2X,7HSTDDEV=,2X
     *  ,E14.4/1X,4HS/N=,2X,E14.4)
        RETURN
        END
```

```
C    QSPLTX.FR      5/10/78          REV. A  11/1/78
C                                    REV. B 11/2/78
C                                    REV. C  11/28/78
C                                    REV. D  12/6/78
C
      SUBROUTINE QSPLT(IS,NC,NR,QFLQ)
C
C     QSPLT MAKES A QREY SCALE PLOT OF THE MAQNITUDE
C     OF THE NR X NC COMPLEX ARRAY IS
C     QFLQ= 0  NEW LINEAR QRAY SCALE
C           1  PREVIOUS QRAY SCALE
C           2  LOQ QRAY SCALE
C           3  NORMALIZE PEAK VALUE TO 99
C           4  NORMALIZE PEAK VALUE TO 1000
      COMMON/QQ/VAL(10),QRAT(30)
      COMMON/JJ/LBUF(132,3)
      DIMENSION KBUF(260),TAB(1024),AVE(64)
      INTEQER TAB,VAL,QRAT,QFLQ
      COMPLEX IS(64,64),DD(64),CPKMQ,CPNORM,CSUM
C
      DATA LBUF(131,1)/5,12K/
      DATA LBUF(131,2)/5,12K/
      DATA LBUF(131,3)/5,12K/
      TYPE "ONE"
C
C
C     INPUT NUMBER OF POINTS TO BE AVERAQED
C     ZERO AVE ARRAY
C
      IF(QFLQ .QT. 3)NAVE=QFLQ-3
      QO TO 333
      ACCEPT "NUMBER OF POINTS AVERAQED",NAVE
  333 CONTINUE
      DO 202 I202=1,64
      AVE(I202)=0.
  202 CONTINUE
C
C
      TYPE"IN QSPLT"
C
      DATA QRAT/100K,100K,122K,111K,122K,155K,4*177K,
     *  100K,122K,100K,144K,155K,3*122K,155K,177K,
     *  100K,100K,122K,111K,122K,155K,4*177K/
C
      CALL OPEN(2,"SCRATCH",0,IER,512)
C
      DO 1 I1=1,260
1     KBUF(I1)=0
C
      DO 23 I1=1,NR
      DO 21 I2=1,NC
21    DD(I2)=IS(I2,I1)
      CALL WRBLK(2,I1-1,DD,1,IER)
      IF(IER.NE.1) STOP --WRBLK ERR--
23    CONTINUE
30    CONTINUE
      TYPE "TWO"
      PKMQ=0
      DO 5 J=1,NR
      DO 5 I=1,NC
      SS=CABS(IS(I,J))
      IF(QFLQ .EQ. 2) SS=10.*ALOQ10(SS+1.)
      IF(SS.QT.PKMQ) PKMQ=SS
```

```
5       CONTINUE
C
        TYPE "THREE"
C       NORMALIZE  INPUT ARRAY
C
        IF(QFLQ .LT. 3)QO TO 87
        PNORM=1000.
        IF(QFLQ .QE. 3)PNORM=99.
        CPNORM=CMPLX(PNORM,0.)
        CPKMQ=CMPLX(PNORM/PKMQ,0.)
C
        DO 88 I88=1,NC
        DO 88 J88=1,NR
        IS(I88,J88)=IS(I88,J88)*CPKMQ
C
C       FIND THE "NAVE" NUMBER OF PEAK POINTS
C
        SS=CABS(IS(I88,J88))
        IF(SS .LE. AVE(NAVE))QO TO 204
        AVE(NAVE)=SS
        NAVEL=NAVE-1
        DO 206 I206=1,NAVEL
        IF(AVE(NAVE+1-I206) .LE. AVE(NAVE-I206))QO TO 204
        TEMP=AVE(NAVE-I206)
        AVE(NAVE-I206)=AVE(NAVE+1-I206)
        AVE(NAVE+1-I206)=TEMP
  206   CONTINUE
  204   CONTINUE
C
C
   88   CONTINUE
        TYPE "FOUR"
C
C
C       CALCULATE STATISTICS OF BACKGROUND
C
   87   CONTINUE
C
        CSUM=CMPLX(0.,0.)
        BSUM=0.
        BSUM2=0.
C
        DO 89 I89=1,NC
        A=CABS(IS(I89,1))
        B=CABS(IS(I89,NR))
        CSUM=CSUM+IS(I89,1)+IS(I89,NR)
        IS(I89,1)=CPNORM
        IS(I89,NR)=CPNORM
        BSUM2=BSUM2+A*A+B*B
   89   CONTINUE
        TYPE "FIVE"
        NROWS=NR-1
C
        DO 91 I91=2,NROWS
        A=CABS(IS(1,I91))
        B=CABS(IS(NR,I91))
        CSUM=CSUM+IS(1,I91)+IS(NC,I91)
        IS(1,I91)=CPNORM
        IS(NC,I91)=CPNORM
        BSUM2=BSUM2+A*A+B*B
   91   CONTINUE
        TYPE "SIX"
C
```

```
;PLPT.SR          04/23/78
;
          .TITLE PLPT
          .ZREL
          .TXTM 1
          .ENT PLPT
          .EXTD .CPYL,.FRET,.STOP
          .NREL
;
;         SUBROUTINE PLPT(LBUF(1,J))
;
;         LBUF IS A 132. WORD BUFFER
LBUF=-167
          1
PLPT:     JSR @.CPYL
          LDA 3,LBUF,3
          STA 3,TEMP
          LDA 0,PASS
          MOV 0,0,SZR
          JMP PASS2
          ISZ PASS
          SUBC 0,0
          STA 0,N
          JMP .+2
LP:       ISZ N
          LDA 0,ABUF
          LDA 2,N             ;CHANNEL #
          .SYSTM
          .CHSTS 77
          JMP .+1
          LDA 0,BUF
          LDA 1,LN
          SUB 0,1,SNR
          JMP PASS2
          DSZ NN
          JMP LP

;         --ERROR, THE $LPT HAS NOT YET BEEN OPENED
;           (IT MUST BE WRITTEN TO BEFORE CALLING PLPT)

          JSR @.STOP
          .TXT /(PLPT) -- $LPT NOT YET OPENED/
;
PASS2:    LDA 1,K132D
          MOVZR 1,1           ;=64
          STA 1,CNT
          LDA 3,TEMP          ;OLD BUFFER POINTER
          LDA 2,APBUF         ;ADDRESS OF PLOT BUFFER
NXT:      LDA 0,0,3
          MOVS 0,0
          INC 3,3
          LDA 1,0,3
          ADD 1,0
          STA 0,0,2
          INC 2,2
          INC 3,3
          DSZ CNT
          JMP NXT
;
          LDA 0,APBUF         ;PLOT BUFFER ADDRESS
          MOVZL 0,0           ;BYTE POINTER
          LDA 1,K132D
          LDA 2,N             ;GET CHANNEL NUMBER
          .SYSTM
          .WRS 77
          JMP .+1
          JSR @.FRET
;
K132D:    132.
TEMP:     0
CNT:      0
PASS:     0
N:        0
NN:       62.
ABUF:     BUF
LN:       "$*400+"L
APBUF:    PBUF      ;ADDRESS OF PLOT BUFFER
BUF:      .BLK 22
PBUF:     .BLK 67.            ;=132/2 + 1 EXTRA
          .END
```

```
; PICPLT.SR        05/08/78
;
; SUBROUTINE PICPLT(DATAO,DATAL,VALO,GRAT,LINEO,TAB)
; THIS ROUTINE OBTAINS A SEGMENT OF A PICTURE SCAN LINE
; ( 260 SAMPLES MAX ) , CONVERTS EACH WORD OF DATA TO
; A 3X3 DOT PATTERN(10 SHADES OF GRAY) AND PLOTSTHE SCAN
; LINE AS THREE ROWS OF DOTS ON THE PRINTRONIX LINE
; PRINTER      THE ROUTINE IS CALLED FROM A FORTRAN PROGRAM
; WITH POINTERS TO FOUR TABLES
;    DATAO -M ADDRESS OF FIRST DATA WORD
;    VALO - ADDRESS OF CONVERSION TABLE WHICH CONVERTS
;       DATA TO GRAYSCALE NUMBERS
;    GRAT - ADDRESS OF CONVERSION TABLE WHICH CONVERTS
;       GRAYSCALE NUMBERS TO DOT PATTERNS
;    LINEO - ADDRESS OF FIRST WORD IN OUTPUT YABLE
;       TAB - DIRECT GRAY TABLE LOOKUP
;
;  ALSO, THE CALLING PROGRAM PASSES DATAL WHICH IS THE
; NUMBER OF DATA WORDS TO BE PLOTTED IN EACH LINE
; THREE PLOT LINES ARE RETURNED FOR EACH INPUT LINE

        TITLE PICPL
        IREL
        TXTM 1
        EXTD  CPYL, FRET
        ENT PICPL
        NREL
DATAO= -167
DATAL=DATAO+1
VALO=DATAL+1
GRAT=VALO+1
LINEO=GRAT+1
TAB=LINEO+1
 FS=TAB-DATAO+1
;
        FS
PICPL   JSR @ CPYL
        LDA 0,@DATAL,3
        STA 0,DATAP
        STA 0,DATP
        LDA 0,DATAO,3
        STA 0,DATAR
        LDA 0,VALO,3
        STA 0,VALP
        LDA 0,GRAT,3
        STA 0,GRATP
        LDA 0,LINEO,3
        STA 0,LINEP
        LDA 0,ABUF
        STA 0,BUFP
        LDA 0,DATAR
        STA 0,DATAI
        LDA 0,TAB,3      ;DIRECT GRAY TABLE POINTER
        STA 0,ATAB
        LDA 0,ATAB
        LDA 3,ABUF
LOOP1   LDA 2,@DATAI
        ADD 0,2  ;FORM OFFSET INTO TAB
        LDA 1,0,2        ;GET VALUE
        STA 1,0,3        ;STORE IN OUTPUT
        ISZ DATAI
        INC 3,3
        DSZ DATP         ;DONE WITH LINE?
        JMP LOOP1
;  THE DATA IS NOW CONVERTED TO GRAYSCALE NUMBERS RANGING
; FROM 1 TO 10 AND IS STORED IN BUF
        LDA 0,ROW        ;ROW LENGTH
        STA 0,ROWP       ;COUNTER
        LDA 0,LINEP      ;LINE POINTER
        STA 0,LINP       ;COUNTER
LOOP3   LDA 0,ABUF       ;INPUT DATA BUFFER
        STA 0,BUFP
        LDA 0,GRATP      ;GRAY TABLE POINTER
        NEG 0,0          ;SUBTRACT 1
        COM 0,0
        STA 0,GRAP
        LDA 0,DATAP      ;DATA LENGTH
        STA 0,DATP       ;COUNTER
        SUBC 0,0         ;=0
        STA 0,PICT1      ;CLEAR PICT1
LOOP4   LDA 0,@BUFP      ;GET A GRAYSCALE
        LDA 1,GRAP
        ADD 0,1          ;ADD TO GRAY TABLE
        STA 1,GRATI
        LDA 0,@GRATI     ;VALUE IN GRAY TABLE
        LDA 1,DATP       ;GET DATA POINT NUMBER
        MOVR@ 1,1,SZC    ;DATP EVEN
        JMP ODD
        LDA 1,MSKEV      ;EVENN MASK
        AND 0,1  ;MASK PROPER NIBBLE
        STA 1,PICT1      ;RIGHT HALF OF OUTPUT WORD
        JMP EVEN
;
ODD     LDA 1,MSKODD     ;ODD MASK
        AND 0,1  ;MASK ODD HALF
        LDA 0,PICT1      ;OLD HALF OF DATA WORD
        ADD 0,1  ;RESULT
        STA 1,@LINP      ;STORE IN OUTPUT ARRAY
        ISZ LINP
EVEN    ISZ BUFP         ;NEXT INPUT
        DSZ DATP         ;DONE WITH ROW?
        JMP LOOP4        ;NO
        LDA 0,LINEP
        LDA 1,LINEL      ;LINE LENGTH
        ADD 0,1
        STA 1,LINEP      ;BUMP LINE POINTER BY 1 LINE LEN
        STA 1,LINP
        LDA 0,GRATP
        LDA 1,OFFST
        ADD 0,1
        STA 1,GRATP
        STA 1,GRAP       ;GO TO NEXT ROW OF DOTS IN TABLE
        DSZ ROWP         ;3 LINES OUTPUT?
        JMP LOOP3
        JSR @ FRET       ;RETURN TO CALLING PROGRAM
;
ATAB    TAB
ABUF    BUF
BUFP    0
CNT2    0
DATAP   0
DATP    0
DATAR   0
VALP    0
GRATP   0
LINEP   0
DATAI   0
ROWP    0
LINP    0
GRAP    0
GRATI   0
LINEL   132
OFFST   10
MSKEV   7
MSKOD   170
PICT1   0
ROW     3
VAP     0
BUF     BLK 260
        END
```

$LPT.	0	RAP	03/27/79	12:01	03/27/79	[000000]	1
$TTI.	0	APW	03/27/79	12:01	03/27/79	[000000]	1
$TTI1.	0	APW	03/27/79	12:01	03/27/79	[000000]	1
$TTO.	0	RAP	03/27/79	12:01	03/27/79	[000000]	1
$TTO1.	0	RAP	03/27/79	12:01	03/27/79	[000000]	1
$TTP.	0	RAP	03/27/79	12:01	03/27/79	[000000]	1
$TTP1.	0	RAP	03/27/79	12:01	03/27/79	[000000]	1
$TTR.	0	APW	03/27/79	12:01	03/27/79	[000000]	1
$TTR1.	0	APW	03/27/79	12:01	03/27/79	[000000]	1
AIDEB.RB		UTILITY:AIDEB.RB					
ASM.SV		UTILITY:ASM.SV					
BREAK.SV	51172	SD	10/31/78	11:41	10/31/78	[015641]	0
CLG.SV		FORT4:CLG.SV					
COM.CM	10		03/27/79	12:06	03/27/79	[015227]	0
CONS.FR	256	D	12/05/78	15:25	01/04/79	[014201]	0
CONS.RB	482	D	12/05/78	15:25	01/03/79	[015572]	0
CR16SCN.FR	2114	D	11/09/78	13:17	01/04/79	[014234]	0
CR16SCN.RB	3288	D	11/09/78	13:18	11/09/78	[015705]	0
CR16SCN.SV	10752	SD	11/09/78	13:22	11/09/78	[014516]	0
CR3.FR	2603	D	12/21/78	16:03	01/04/79	[015452]	0
CR3.RB	3882	D	12/21/78	16:04	12/21/78	[015216]	0
CR3.SV	11264	SD	12/21/78	16:05	12/21/78	[016077]	0
CR4.FR	2240	D	12/22/78	10:25	12/22/78	[015633]	0
CR4.RB	3256	D	12/22/78	10:26	12/22/78	[015321]	0
CR4.SV	10752	SD	12/22/78	10:26	12/22/78	[014146]	0
CR4SCN.FR	1669	D	11/09/78	10:11	01/04/79	[015610]	0
CR4SCN.RB	2400	D	11/09/78	10:12	11/09/78	[015313]	0
CR4SCN.SV	10240	SD	11/09/78	10:14	11/09/78	[014112]	0
CR5.FR	2603	D	12/22/78	10:00	12/22/78	[013060]	0
CR5.RB	3882	D	12/22/78	10:01	12/22/78	[015425]	0
CR5.SV	11264	SD	12/22/78	10:02	12/22/78	[014376]	0
CR6.FR	2697	D	12/22/78	10:10	12/22/78	[014207]	0
CR6.RB	4046	D	12/22/78	10:11	12/22/78	[015577]	0
CR6.SV	11264	SD	12/22/78	10:11	12/22/78	[014457]	0
CRLNOI.FR	1847	D	11/10/78	08:39	01/04/79	[015440]	0
CRLNOI.RB	2516	D	11/10/78	08:39	11/10/78	[015154]	0
CRLNOI.SV	11264	SD	11/10/78	08:40	11/10/78	[016023]	0
CRN.FR	1883	D	12/13/78	15:29	12/13/78	[014245]	0
CRN.RB	2644	D	12/13/78	15:30	12/13/78	[015715]	0
CRN.SV	10240	SD	12/13/78	15:30	12/13/78	[014550]	0
CRNOI.FR	1530	D	11/10/78	08:17	01/04/79	[015137]	0
CRNOI.RB	2108	D	11/08/78	12:09	11/08/78	[014724]	0
CRNOI.SV	10752	SD	11/08/78	12:13	11/08/78	[015331]	0
CRSCN.FR	1560	D	11/03/78	09:45	01/04/79	[014637]	0
CRSCN.RB	2186	D	11/03/78	09:45	12/01/78	[014365]	0
CRSCN.SV	9728	SD	12/01/78	09:53	12/01/78	[014765]	0
CRSQ.FR	1835	D	12/21/78	08:02	01/04/79	[015047]	0
CRSQ.RB	2512	D	12/21/78	08:03	12/21/78	[014644]	0
CRSQ.SV	10240	SD	12/21/78	08:03	12/21/78	[015171]	0
CRSUM.FR	258	D	11/03/78	12:21	01/04/79	[014762]	0
CRSUM.RB	496	D	11/03/78	12:21	11/03/78	[014601]	0
CRSUM.SV	9216	SD	11/03/78	12:22	11/03/78	[015114]	0
CRW.FR	2603	D	12/22/78	08:38	12/22/78	[015527]	0
CRW.RB	3882	D	12/22/78	08:39	12/22/78	[015271]	0
CRW.SV	11264	SD	12/22/78	08:40	12/22/78	[013415]	0
DIFF.FR	589	D	12/06/78	14:07	01/04/79	[014544]	0
DIFF.RB	838	D	12/06/78	14:07	01/03/79	[014175]	0
EDIT.SV		UTILITY:EDIT.SV					
ESAC3.RB		UTILITY:ESAC3.RB					
F5ENV.LB		FORT:F5ENV.LB					
F5IO.LB		FORT:F5IO.LB					
F5ISA.LB		FORT:F5ISA.LB					
F5MATH.LB		FORT:F5MATH.LB					
F5TASK.LB		FORT:F5TASK.LB					
FAD.RB	1472	D	10/09/78	08:42	10/19/78	[014273]	0
FASTNCAL.RB		FORT:FASTNCAL.RB					
FASTPCAL.RB		FORT:FASTPCAL.RB					
FBDRVT.FR	5842	D	01/03/79	08:53	01/04/79	[016127]	0
FBDRVT.MC	97	D	12/22/78	08:28	01/04/79	[014722]	0
FBDRVT.RB	8372	D	01/03/79	08:55	01/03/79	[015376]	0
FBDRVT.SV	25088	SD	01/03/79	08:56	01/03/79	[014303]	0
FCOM.CM	15		01/03/79	15:28	01/03/79	[015536]	0
FDUMP.SV		UTILITY:FDUMP.SV					
FFT.RB	2120	D	12/15/78	08:10	12/15/78	[015064]	0
FFT2D.RB	2130	D	10/05/78	17:09	01/03/79	[014604]	0
FIV.SV		FORT4:FIV.SV					
FLOAD.SV		UTILITY:FLOAD.SV					
FMT.LB		FORT4:FMT.LB					
FNORM.FR	311	D	12/07/78	13:11	01/04/79	[014615]	0
FNORM.RB	560	D	12/07/78	11:20	01/03/79	[014242]	0
FORT.LB		FORT4:FORT.LB					
FORT.SV		FORT4:FORT.SV					
FORTRAN.ER		FORT:FORTRAN.ER					
FORTRAN.SV		FORT:FORTRAN.SV					
FQPF.RB	2818	D	10/16/78	13:50	10/19/78	[014434]	0
FQPO.RB	674	D	10/05/78	16:20	10/19/78	[014612]	0
FQPXY.RB	3728	D	10/16/78	12:45	10/19/78	[014626]	0
FSYM.RB	1534	D	10/09/78	08:34	10/19/78	[013553]	0

```
FTC_TX                          FORT FTC_TX
FTC1_SV                         FORT FTC1_SV
FTC2_SV                         FORT FTC2_SV
FTC3_SV                         FORT FTC3_SV
FTC4_SV                         FORT FTC4_SV
FTC5_SV                         FORT FTC5_SV
FTC6_SV                         FORT FTC6_SV
FTC7_SV                         FORT FTC7_SV
FTCE_SV                         FORT FTCE_SV
QSPLT_FR          4436  D       11/28/78 10:21  01/04/79  [013540]      0
QSPLT_RB          6498  D       11/28/78 10:22  11/28/78  [015511]      0
QSPLTX_FR         4532  D       12/06/78 15:14  01/04/79  [014221]      0
QSPLTX_RB         6576  D       12/06/78 15:15  01/03/79  [013615]      0
QSPO_RB            404  D       10/09/78 09:49  10/19/78  [014670]      0
HADD_FR            498  D       12/06/78 14:19  01/04/79  [014624]      0
HADD_RB            838  D       12/06/78 14:20  01/03/79  [014266]      0
HCRE_FR            349  D       12/06/78 14:16  01/04/79  [013054]      0
HCRE_RB            846  D       12/06/78 14:17  01/03/79  [014652]      0
HDRV_FR           1403  D       10/31/78 12:27  01/04/79  [013307]      0
HDRV_MC             37  D       10/31/78 12:30  11/03/78  [014374]      0
HDRV_RB           1856  D       10/31/78 12:27  11/03/78  [013072]      0
HDRV_SV          13312  SD      11/03/78 15:35  11/03/78  [013537]      0
HRCON_FR           496  D       12/06/78 14:23  01/04/79  [013062]      0
HRCON_RB           838  D       12/06/78 14:24  01/03/79  [014665]      0
HSDRV_FR          1543  D       11/03/78 10:32  01/05/79  [014656]      0
HSDRV_RB          2102  D       11/03/78 10:41  11/03/78  [014426]      0
HSDRV_SV         13824  SD      11/03/78 12:51  11/03/78  [013012]      0
IMAGER_FR          235  D       11/03/78 13:24  01/04/79  [013060]      0
IMAGER_RB          488  D       11/03/78 13:24  11/03/78  [014663]      0
IMAGER_SV        16384  SD      11/03/78 13:25  11/03/78  [013230]      0
IREAD_FR          1599  D       10/31/78 11:45  01/04/79  [013357]      0
IREAD_RB          2322  D       10/31/78 11:45  01/03/79  [013105]      0
LEFD_RB                         FORT LEFD_RB
LEFE_RB                         FORT LEFE_RB
LFE_SV                          UTILITY LFE_SV
LINK_LB                         VPLOT LINK_LB
LONGTRACE_RB                    FORT LONGTRACE_RB
MAC_PS                          UTILITY MAC_PS
MAC_SV                          UTILITY MAC_SV
MACXR_SV                        UTILITY MACXR_SV
MAP_DR            1222  APWC    02/23/79 14:37  03/27/79  [000017]      0
MAPP_LB                         VPLOT MAPP_LB
MATH_LB                         UTILITY MATH_LB
MHOLD_FR           939  D       11/03/78 15:33  01/04/79  [013574]      0
MHOLD_RB          1292  D       11/03/78 15:33  11/28/78  [013302]      0
MHOLD_SV         17408  SD      11/28/78 10:38  11/28/78  [013572]      0
MULT_FR           1841  D       12/05/78 08:25  01/04/79  [013445]      0
MULT_RB           8454  D       12/05/78 08:25  12/05/78  [015162]      0
MULT_SV          10240  SD      12/05/78 08:26  12/05/78  [016052]      0
NOISE_FR          1446          10/24/78 08:44  01/04/79  [015373]      0
NOTRACE_RB                      FORT NOTRACE_RB
NPLPT_RB           326  D       12/15/78 16:19  12/15/78  [014262]      0
NPLPT_SR          1088  D       12/15/78 16:18  12/15/78  [014716]      0
OEDIT_SV                        UTILITY OEDIT_SV
OVLDR_SV                        UTILITY OVLDR_SV
PAR6464_FR        1013  D       11/01/78 12:36  01/04/79  [013420]      0
PAR6464_RB        1322  D       11/01/78 12:36  11/01/78  [015143]      0
PAR6464_SV       14848  SD      11/01/78 12:37  11/01/78  [015724]      0
PARNOR_SV            0  D       11/02/78 11:15  11/02/78  [015011]      0
PARNORM_FR         335  D       11/01/78 14:34  01/04/79  [013436]      0
PARNORM_RB         396  D       11/01/78 14:34  11/02/78  [015152]      0
PARNORM_SV       15360  SD      11/02/78 13:06  11/02/78  [015764]      0
PICPLT_RB          496  D       05/08/78 00:06  01/03/79  [015364]      0
PLPT_RB            576  D       04/23/78 00:58  01/03/79  [014300]      0
PPEP_LB                         VPLOT PPEP_LB
PPEP1200_LB                     VPLOT PPEP1200_LB
RAND_RB            122  D       08/30/78 11:04  11/10/78  [015371]      0
RAND_SR            181  D       04/18/78 03:21  09/24/78  [014204]      0
RASL_SV                         VPLOT RASL_SV
RASM_SV                         VPLOT RASM_SV
RASS_SV                         VPLOT RASS_SV
RBLC_FR            367  D       12/06/78 09:21  01/04/79  [013423]      0
RBLC_RB            640  D       12/06/78 09:21  01/03/79  [015147]      0
RBLKC_FR           463  D       11/10/78 16:20  01/04/79  [013762]      0
RBLKC_RB           764  D       10/24/78 11:32  11/28/78  [015366]      0
RGAUS_FR          1223  D       08/29/78 15:40  11/06/78  [015100]      0
RGAUS_RB           634  D       08/29/78 15:30  11/10/78  [014676]      0
RLDR_OL                         UTILITY RLDR_OL
RLDR_SV                         UTILITY RLDR_SV
RRASL_SV                        VPLOT RRASL_SV
RRASM_SV                        VPLOT RRASM_SV
RRASS_SV                        VPLOT RRASS_SV
SCRAM_FR          1646  D       11/07/78 14:31  01/04/79  [014617]      0
SCRAM_RB          2296  D       11/07/78 14:31  11/07/78  [014654]      0
SCRAM_SV         11776  SD      11/07/78 14:32  11/07/78  [014732]      0
SDEF_RB           2494  D       10/23/78 14:14  10/23/78  [014444]      0
SDEFO_RB           404  D       10/09/78 09:43  10/19/78  [014576]      0
SHORTRTER_RB                    FORT SHORTRTER_RB
SORT_LB                         VPLOT SORT_LB
SORT_SV                         VPLOT SORT_SV
SPEED_ER                        UTILITY SPEED_ER
SPEED_SV                        UTILITY SPEED_SV
SYS_DR           27136  APWDV   03/27/79 12:01  03/27/79  [007252]      1
SYS_LB                          UTILITY SYS_LB
THREAD_SV                       VPLOT THREAD_SV
VRASL_SV                        VPLOT VRASL_SV
VRASM_SV                        VPLOT VRASM_SV
VRASS_SV                        VPLOT VRASS_SV
WBLC_FR            661  D       12/15/78 08:32  01/05/79  [014513]      0
WBLC_RB           1038  D       12/06/78 09:25  01/03/79  [014141]      0
WBLKC_FR           724  D       10/31/78 12:28  01/05/79  [014672]      0
WBLKC_RB          1174  D       10/31/78 12:28  11/10/78  [014453]      0
XREF_SV                         UTILITY XREF_SV
```

LIST OF DATA FILES ON TAPE

BX.	33280	D	
F1010.	33280	D	These files are four point images at the specified coordinates. They can be read using IMAGER-SV for grayscale plot or by SUBROUTINE SDEF (in file IREAD,FR) for input into memory.
F1040.	33280	D	
F1545.	33280	D	
F2032.	33280	D	
F2432.	33280	D	
F2828.	33280	D	
F2832.	33280	D	
F2836.	33280	D	
F3228.	33280	D	
F3232.	33280	D	
F3236.	33280	D	
F3240.	33280	D	
F3632.	33280	D	
F3636.	33280	D	
F4032.	33280	D	
F4040.	33280	D	
F6060.	33280	D	
FDRV.	95	D	
FF1010.	33280	D	These files are Fourier transforms of the above files. They can be read into memory using RBLC, or TBLKC subroutines.
FF1040.	33280	D	
FF2032.	33280	D	
FF2432.	33280	D	
FF2828.	33280	D	
FF2832.	33280	D	
FF2836.	33280	D	
FF3228.	33280	D	
FF3232.	33280	D	
FF3236.	33280	D	
FF3240.	33280	D	
FF3632.	33280	D	
FF3636.	33280	D	
FF4032.	33280	D	
FF4040.	33280	D	
FF6060.	33280	D	
FFLNOI3.	33280	D	
FIMAGEN.	33280	D	
FIMAGEW.	33280	D	
FLIB.		FORT:FLIB.	
FLN20.	33280	D	These files are Fourier transforms of random noise files.
FLNOI1.	33280	D	
FLNOI2.	33280	D	
FLNOI3.	33280	D	
FMREF1.	33280	D	
FMREF2.	33280	D	
FMREF3.	33280	D	
FMREF4.	33280	D	
FMREF5.	33280	D	
FMREF6.	33280	D	
FMREF7.	33280	D	
FMREF8.	33280	D	
FNOISE.	33280	D	
FNOISE1.	33280	D	
FNOISE2.	33280	D	
FNOISE3.	33280	D	
FNOISE4.	33280	D	
FNOISE5.	33280	D	
FTA1.	33280	D	
FTA2.	33280	D	
FTA3.	33280	D	
FTID1.	33280	D	
FTID2.	33280	D	
FTID3.	33280	D	
HRP2020.	33280	D	

HRP4025.	33280	D	
HRPOINT.	33280	D	
HRT4025	33280	D	
HX.	33280	D	
IMAGEN.	33280	D	Image of N&W
IMAGEW.	33280	D	
LIM16.	33280	D	Limited Random Noise files, read using IMAGER or SUB SDEF.
LIM32.	33280	D	
LIM64.	33280	D	
LIM8.	33280	D	
LN16.	33280	D	
LN18.	33280	D	
LN20.	33280	D	
LN22.	33280	D	
LN24.	33280	D	
LN26.	33280	D	
LN28.	33280	D	
LN30.	33280	D	
LN32.	33280	D	
LP16H1	33280	D	
LP16H2	33280	D	
MAOI.0	50	D	
MREF1.	33280	D	Random noise scenes, read using IMAGER or SUB SDEF.
MREF2	33280	D	
MREF3.	33280	D	
MREF4.	33280	D	
MREF5.	33280	D	
MREF6.	33280	D	
MREF7.	33280	D	
MREF8.	33280	D	
NOISE.	33280	D	
NOISE1	33280	D	
NOISE10.	33280	D	
NOISE11	33280	D	
NOISE12	33280	D	
NOISE13.	33280	D	
NOISE14	33280	D	
NOISE15	33280	D	
NOISE16.	33280	D	
NOISE2.	33280	D	
NOISE3.	33280	D	
NOISE4	33280	D	
NOISE5.	33280	D	
NOISE6	33280	D	
NOISE7	33280	D	
NOISE8.	33280	D	
NOISE9	33280	D	
P0527.	33280	D	Single point images, read using IMAGER or SDEF.
P1010.	33280	D	
P1535	33280	D	
P1536.	33280	D	
P1545.	33280	D	
P1635	33280	D	
P1636.	33280	D	
P16H1.	33280	D	
P16H2	33280	D	
P16H4.	33280	D	
P16H8.	33280	D	
P1H16.	33280	D	
P1H8.	33280	D	
P2020.	33280	D	
P2351	33280	D	
P2540	33280	D	
P3030	33280	D	
P3313.	33280	D	
P3327	33280	D	
P3339	33280	D	
P4025.	33280	D	
P4040.	33280	D	
P4313.	33280	D	
P4315	33280	D	
P4317.	33280	D	
P4H1	33280	D	
P4H16.	33280	D	
P4H2.	33280	D	
P4H4.	33280	D	

```
P4H8.          33280  D
P5050.         33280  D
P6060.         33280  D
S2824.         33280  D
S2828.         33280  D
S2832.         33280  D
S2836.         33280  D
S2840.         33280  D
S3224.         33280  D
S3228.         33280  D
S3232.         33280  D
S3236.         33280  D
S3240.         33280  D
S3244.         33280  D
S3624.         33280  D
S3628.         33280  D
S3632.         33280  D
S3636.         33280  D
S3640.         33280  D
SCRAM1.        33280  D
SCRAM10.       33280  D
SCRAM11.       33280  D
SCRAM12.       33280  D
SCRAM13.       33280  D
SCRAM14.       33280  D
SCRAM15.       33280  D
SCRAM16.       33280  D
SCRAM2.        33280  D
SCRAM3.        33280  D
SCRAM4.        33280  D
SCRAM5.        33280  D
SCRAM6.        33280  D
SCRAM7.        33280  D
SCRAM8.        33280  D
SCRAM9.        33280  D
SQ3216.        33280  D
SQ3224.        33280  D
SQ3232.        33280  D
SQ32.          33280  D
T4045.         33280  D
TAN135A.       33280  D
TAN135B.       33280  D
TANK90.        33280  D
TANK10.        33280  D
TANK135.       33280  D
TANK180.       33280  D
TANK225.       33280  D
TANK45.        33280  D
TANK90.        33280  D
TANKTES.       33280  D
TFLIB.                FORT.TFLIB.
TID1.          33280  D
TID2.          33280  D
TID3.          33280  D
TID4.          33280  D
TID5.          33280  D
TID6.          33280  D
WA1.           33280  D
WA2.           33280  D
WB1.           33280  D
WB2.           33280  D
WC1.           33280  D
WC2.           33280  D
```

16 point images*

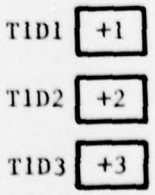
Random noise images

Square images*

Tank images, number in filename is tank orientation. Ex TAN 90 is tank at 90°

Tank I.D. files

Dummy files*



*Read using IMAGER or SUB SDEF (in IREAD. FR).

APPENDIX A

FEATURE EXTRACTION BY HOLOGRAPHY

We are presented with the problem where we wish to use a hologram to distinguish between two similar reference patterns, $\rho_a(x)$ and $\rho_b(x)$. Because the patterns are similar, we may say they are partially correlated. We may also express the similarity by saying the two patterns share a common subset. The measure of similarity of the two patterns is given by the correlation coefficient:

$$C_{ab} = \frac{\int \rho_a(x)\rho_b(x)\, dx}{\sqrt{\int \rho_a^2(x)dx \int \rho_b^2(y)dy}} \qquad \text{(A-1)}$$

In terms of the Fourier transformed patterns we may write

$$C_{ab} = \frac{\int R_a^*(\xi)\, R_b(\xi)\, d\xi}{\sqrt{\int |R_a|^2 d\xi \cdot \int |R_b|^2 d\xi}} \qquad \text{(A-2)}$$

We recognize that $|R_a|^2$ and $|R_b|^2$ are constants, by condition of the experiment.

The numerator of Equation A-2 can also be recognized as the integral over a constant. The argument for this is as follows: Reconstruction of a hologram made by reference $\rho_a(x)$ depends on the similarity of reconstruction reference $\rho_b(x)$. The similarity may be denoted by the correlation between these two functions. If we treat the functions as vectors in a space spanned by the independent variable, x, we may represent the process as in Figure A-1:

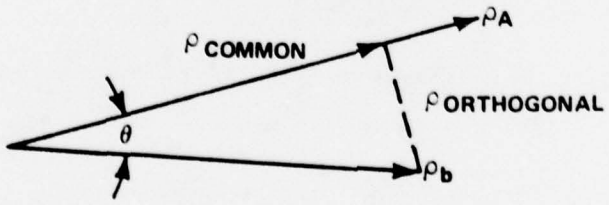


FIGURE A-1. VECTOR REPRESENTATION OF CORRELATION

We see that ρ_b may be resolved into two vectors. One vector lies along the original reference and is the correlated part of ρ_b. We can arbitrarily define this correlated vector as the subset of ρ_b which is common to both ρ_a and ρ_b. The other vector which is a component of ρ_b is orthogonal to ρ_a. Thus, in the reconstruction process we have a term like:

$$\rho_a \circledast \rho_a * \sigma = (\rho_{common} \circledast \rho_a + \rho_{orthogonal} \circledast \rho_a) * \sigma$$
$$= (\text{constant}) \times \sigma + (\text{noise}) \circledast \sigma \qquad \text{(A-3)}$$

We get the reconstruction because $\rho_{common} \circledast \rho_a$ must meet the reconstruction condition of being delta like. This means that the Fourier transform of this term must be a constant. We can therefore write:

$$\rho_b \circledast \rho_a = (\rho_{com} \circledast \rho_a + \rho_{orth} \circledast \rho_a) \xrightarrow{F} R_{com} R_a^* + R_{orth} R_a^* \qquad \text{(A-4)}$$

Thus, we may write

$$R_a^* R_b = R_b^* R_a = \text{constant}. \qquad \text{(A-5)}$$

Since we have integrals over constants, the area of the recording cancels out from the numerator and denominator of Equation A-2. Thus, Equation A-2 is equivalent to

$$C_{ab} = \frac{R_a^* R_b}{|R_a| |R_b|} = \frac{R_b^* R_a}{|R_a| |R_b|} = \text{correlation coefficient between a + b} \qquad \text{(A-6)}$$

The correlation between the two similar patterns ρ_a and ρ_b may, by an appropriate twist of Equation A-6, be put to use.

Suppose we wish to form a multiplexed hologram such that advent of the reference pattern ρ_a will cause a unique reconstruction of the associated object pattern σ_a. Similarly, ρ_b will reconstruct only σ_b.

An ordinary multiplexed hologram will lead to a false reconstruction. For example, the hologram

$$H = R_a^* S_a + R_b^* S_b \qquad \text{(A-7)}$$

will produce the output

$$R_a H = |R_a|^2 S_a + R_a R_b^* S_b \qquad \text{(A-8)}$$

when the reference ρ_a is incident. Equation A-8 may be interpreted, with the aid of Equation A-6 as:

$$R_a H = |R_a|^2 S_a + |R_a||R_b| C_{ab} \cdot S_b \qquad \text{(A-9)}$$

We see that the unwanted output S_b is reconstructed in proportion to the degree of correlation between ρ_a and ρ_b.

If, however, we make a hologram which includes subtractive terms it becomes possible to cancel the false reconstruction.

Such a hologram would take the following form:

$$H = R_a^* S_a + R_b^* S_b - k_1 R_b^* S_a - k_2 R_a^* S_b \qquad \text{(A-10)}$$

In Equation A-10 k_1 and k_2 are constants. We evaluate these constants by performing a reconstruction.[2] Thus, with ρ_a incident we get the output

$$R_a H = (|R_a|^2 - k_1 R_a R_b^*) S_a + (R_a R_b^* - k_2 |R_a|^2) S_b \qquad \text{(A-11)}$$

If k_2 is adjusted so that there is no output of the false reconstruction, S_b, it will take the value:

$$k_2 = \frac{R_a R_b^*}{|R_a|^2} = \frac{|R_b|}{|R_a|} C_{ab}, \text{ by Equation (A-6).} \qquad \text{(A-12)}$$

By a symmetrical experiment we find that

$$k_1 = \frac{R_a R_b^*}{|R_b|^2} = \frac{|R_a|}{|R_b|} C_{ab} \qquad \text{(A-13)}$$

gives no output of S_a when reference ρ_b is incident.

By proper adjustment of k_1 and k_2 the hologram may be tuned to give unique outputs when either of the two reference patterns are introduced. There is, however a penalty: The desired reconstruction is somewhat reduced. To see the magnitude of this effect we introduce Equations A-12 and A-13 into the reconstruction Equation A-11. The result is

$$R_a H = \left(|R_a|^2 - \frac{(R_a R_b^*)^2}{|R_b|^2} \right) S_a + \left(R_a R_b^* - \frac{R_a R_b^* |R_a|^2}{|R_a|^2} \right) S_b \qquad \text{(A-14)}$$

$$\boxed{R_a H = |R_a|^2 (1 - C_{ab}^2) S_a} \qquad \text{(A-15)}$$

Similarly,

$$R_b H = |R_b|^2 (1 - C_{ab}^2) S_b \qquad \text{(A-16)}$$

In the treatment of the correlation as a geometrical process, the correlation coefficient is simply the cosine of the angle between the abstract vectors $\rho_a(x)$ and $\rho_b(x)$. We see from Equation A-15 that the reconstruction is proportional to the square of the sine between these two vectors.

That is, if the reference functions are orthogonal, the reconstruction will be maximal. Another way of interpreting this is to say that only the orthogonal (uncorrelated) portions participate as references in the reconstruction.

The process outlined above is analogous to the Gram-Schmidt orthogonalization procedure familiar from mathematics. However, there is the substantial difference that the orthogonalization is virtual. The reconstructed patterns do not need to be orthogonal and no physical rotations of the reference patterns take place.

The ability of the hologram to perform orthogonalizations on sets of patterns shows that its power in pattern recognition is substantially greater than previously suspected.

APPENDIX B

EXISTENCE PROOF

Our purpose here is to show that a multiplexed hologram may be formed from two separate subject-reference pairs in such a way that reconstruction of either subject will be free of crosstalk and other clutter. In particular, it will be shown that this is possible even when the two reference patterns are highly correlated.

We start the proof by establishing two reference patterns $R_1(\xi)$ and $R_2(\xi)$. Each of these patterns are broken up into two subsets. One of the subsets will be used to reconstruct the associated subject, and the other will cancel out the crosstalk from the other hologram of the ensemble. Thus, we write:

$$R_1 = R_1' + R_{1c} \tag{B-1}$$

$$R_2 = R_2' + R_{2c} \tag{B-2}$$

Here we have designated R_1' as the subject reconstructor and R_{1c} as the subset which cancels the effects of partial correlations between R_1 and R_2. Similar definitions hold for the division of R_2.

In forming the hologram we pair R_1 with a subject S_1 and R_2 with S_2. We assume adaptation according to feedback principles discussed in the main text. Thus the terms in the multiplexed hologram will reach an equilibrium state resembling that of Equation B-3. We may write the multiplexed hologram as the summation:

$$H = \frac{S_1}{R_1'} + \frac{S_2}{R_2'} \tag{B-3}$$

In order for all crosstalk to be cancelled, the conditions

$$R_1 H = S_1 \tag{B-4}$$

and

$$R_2 H = S_2 \tag{B-5}$$

must be satisfied.

Then, using the definition B-1 in Equation B-3, we get:

$$R_1 H = \left(R_1' + R_{1c}\right)\left(\frac{S_1}{R_1'} + \frac{S_2}{R_2'}\right) = S_1 + \frac{R_1' S_2}{R_2'} + R_{1c} H \tag{B-6}$$

By constraining this result according to Equation B-4 we must have

$$\frac{R_1' S_2}{R_2'} + R_{1c} H = 0 \tag{B-7}$$

By a similar argument we derive:

$$\frac{R_2' S_1}{R_1'} + R_{2c} H = 0 \tag{B-8}$$

If we eliminate H between Equations B-7 and B-8, we find

$$\frac{R_1'^2}{R_2'^2} = \frac{R_{1c} S_1}{R_2 S_2} \tag{B-9}$$

or:

$$\boxed{\frac{R_1'}{R_2'} = \sqrt{\frac{R_{1c} S_1}{R_{2c} S_2}}} \tag{B-10}$$

Next we substitute Equation B-10 into B-2 to get:

$$H = \frac{S_1}{R_1'} + \frac{S_2}{R_1'} \sqrt{\frac{R_{1c} S_1}{R_{2c} S_2}} \tag{B-11}$$

Factoring terms, we have:

$$H = \frac{1}{R_1'} \sqrt{\frac{S_1}{R_{2c}}} \left[\sqrt{S_1 R_{2c}} + \sqrt{R_{1c} S_2} \right] \tag{B-12}$$

By a symmetrical derivation or by employing Equations B-10 in B-12, we can show that H also takes the form

$$H = \frac{1}{R_2'} \sqrt{\frac{S_2}{R_{1c}}} \left[\sqrt{S_1 R_{2c}} + \sqrt{R_{1c} S_2} \right] \tag{B-13}$$

We can test this result by attempting a reconstruction. Playing the reference of Equation B-1 through the hologram of B-12, we get

$$R_1 H = \left(R_1' + R_{1c}\right) \frac{1}{R_1'} \sqrt{\frac{S_1}{R_{2c}}} \left[\sqrt{S_1 R_{2c}} + \sqrt{R_{1c} S_2}\right] \tag{B-14}$$

$$R_1 H = \sqrt{\frac{S_1}{R_{2c}}} \cdot \sqrt{S_1 R_{2c}} + \sqrt{\frac{S_1 S_2 R_{1c}}{R_{2c}}} + R_{1c} H \tag{B-15}$$

$$R_1 H = S_1 + \sqrt{\frac{S_1 S_2 R_{1c}}{R_{2c}}} + R_{1c} H \tag{B-16}$$

Let us examine the term $R_{1c}H$. By equation B-7 we have:

$$R_{1c} H = -\frac{R_1' S_2}{R_2'} \tag{B-17}$$

Inserting B-10 into B-17 we have

$$R_{1c} H = -\sqrt{\frac{R_{1c} S_1 S_2}{R_{2c}}} \tag{B-18}$$

Thus we see, according to B-18 that the last two terms in B-16 cancel. This leaves the desired result:

$$\boxed{R_1 H = S_1} \tag{B-19}$$

By a similar argument we can also show that:

$$\boxed{R_2 H = S_2} \tag{B-20}$$

What we have done here is to demonstrate that there are no pathologies required to form a hologram which will result in the reconstructions of Equations B-19 and B-20. An important point to note is that this result is only possible if the hologram is formed from ratios of subjects and references. This implies a new type of orthogonalization process is operative here. It is not clear, at this time, that the results can be generalized to multiplex ensembles consisting of three or more subholograms.

There are interesting symmetries that come about when it is assumed that the cancellation subsets of the two references are in fact the mutually correlated (or common) portions of these two patterns. Then we have

$$R_{1c} = R_{2c} = R_c \tag{B-21}$$

Then the first simplification comes in Equation B-10, which produces, after cancellation of terms:

$$\frac{R_1'}{R_2'} = \sqrt{\frac{S_1}{S_2}} \tag{B-22}$$

We get the second simplification by eliminating R_2' between Equations B-7 and B-8. The simultaneous solution of these two equations leads to

$$R_1' \ (R_{1c}R_{2c}H^2 - S_1S_2) = 0 \tag{B-23}$$

If R_1' is to be nonzero, we must have

$$H = \sqrt{\frac{S_1S_2}{R_{1c}R_{2c}}} \tag{B-24}$$

In the case where R_{1c} equals R_{2c}, Equation B-24 takes the form

$$H = \frac{\sqrt{S_1S_2}}{R_c} \tag{B-25}$$

In the very special case where $S_1 = S_2 = S$ we have:

$$H = \frac{S}{R_c} \tag{B-26}$$

Thus, the basic ratio form is preserved throughout all these processes.

Ford Aerospace &
Communications Corporation
Aeronutronic Division

Ford Road
Newport Beach, California 92663